Titre: Title:	Optimisation de l'empreinte carbone dans un environnement intercloud : modèles et méthodes
Auteur: Author:	Valérie Danielle Justafort
Date:	2015
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Justafort, V. D. (2015). Optimisation de l'empreinte carbone dans un environnement intercloud : modèles et méthodes [Ph.D. thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/2029/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/2029/
Directeurs de recherche: Advisors:	Samuel Pierre, & Ronald Beaubrun
Programme: Program:	Génie informatique

UNIVERSITÉ DE MONTRÉAL

OPTIMISATION DE L'EMPREINTE CARBONE DANS UN ENVIRONNEMENT INTERCLOUD: MODÈLES ET MÉTHODES

VALÉRIE DANIELLE JUSTAFORT
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE INFORMATIQUE)
NOVEMBRE 2015

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

OPTIMISATION DE L'EMPREINTE CARBONE DANS UN ENVIRONNEMENT INTERCLOUD: MODÈLES ET MÉTHODES

présentée par : JUSTAFORT Valérie Danielle
en vue de l'obtention du diplôme de : Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de :

Mme BELLAÏCHE Martine, Ph. D., présidente
M. PIERRE Samuel, Ph. D., membre et directeur de recherche
M. BEAUBRUN Ronald, Ph. D., membre et codirecteur de recherche
M. QUINTERO Alejandro, Doctorat, membre
M. AJIB Wessam, Ph. D., membre externe

DÉDICACE

À mes parents, Danielle et Serge,
pour votre amour et votre support inconditionnels...

À ma soeur et mon frère, Sybille et Didier,
mes complices de chaque jour...

...Je vous aime.

REMERCIEMENTS

Mes remerciements s'adressent, en tout premier lieu, à mon directeur de recherche, M. Samuel Pierre, et à mon co-directeur, M. Ronald Beaubrun, pour la qualité de leur encadrement, leur disponibilité et leur support tant académique que financier. Leurs conseils avisés et leur degré d'exigence m'ont souvent amenée à peaufiner la justesse de mon raisonnement et ont été pour beaucoup dans la réalisation de ce travail de recherche. Je suis particulièrement reconnaissante à M. Pierre, pour la confiance qu'il a su placer en moi, et ceci, bien avant que j'entame mon parcours doctoral, et pour m'avoir offert cette possibilité de travailler sous sa direction. Un merci particulier à M. Beaubrun pour avoir consenti à effectuer plusieurs traversées, Québec-Montréal, afin de faciliter les échanges.

Je remercie également Mme Martine Bellaïche, M. Alejandro Quintero et M. Wessam Ajib pour l'intérêt porté à mon travail en acceptant de participer à ce jury.

J'adresse mes remerciements aux membres du Département de Génie Informatique et Génie Logiciel (GIGL), aussi bien anciens qu'actuels, et particulièrement à Sabine Kébreau, Micheline Lafrenière et Louise Longtin, pour leurs conseils et leur encouragement tout au long de ces années d'études.

Mes remerciements s'étendent à tous mes collègues du LAboratoire de recherche en Réseautique et Informatique Mobile (LARIM), pour avoir su créer, au laboratoire, un environnement où la bonne humeur, la solidarité, la fraternité et le souci d'entraide prévalent au quotidien. Je remercie particulièrement Georges Abou-Khalil, Hazem Ahmed, Grégory Charlot, Marième Diallo, Éric Fafolahan, Ronald Jean-Julien, Eunice A. Lemamou, Saïda Maaroufi, Moussa Ouédraogo, Aurel Randolph et Germine Séide, pour avoir été, d'une façon ou d'une autre, une source d'inspiration pour moi. Qu'il me soit pardonné de ne pas vous avoir tous nominativement cités ici. Un merci spécial à Georges Abou-Khalil pour ses nombreux conseils, son support et sa grande disponibilité.

Un grand merci à ma famille, à mes ami(e)s et à tous ceux et celles qui, d'une manière ou d'une autre, ont contribué à l'aboutissement de cette thèse. Une mention toute particulière à Mireille et Maurice Jean-Paul, pour m'avoir soutenue et conseillée durant toutes ces années. Sans leur assistance, cette belle aventure n'aurait sans doute jamais débuté !

À ma soeur, Sybille, et mon frère, Didier, un grand merci pour avoir toujours cheminé à mes côtés, pour votre patience envers mes absences souvent répétées, votre affection et votre amour au quotidien. Merci pour cette grande complicité qui nous unit : vos nombreuses taquineries ont toujours égayé mes journées et m'ont permis d'avoir une perception moins austère de la recherche (mon travail porte sur le Cloud, certes, mais je ne vais pas au LARIM pour uniquement faire des petits nuages !). Aussi, un grand merci à toi Syb, pour m'avoir corrigé les épreuves écrites.

Et pour terminer, j'adresse mes profonds et sincères remerciements à mes parents, Danielle et Serge, pour m'avoir donné la VIE et pour les innombrables sacrifices consentis, dans le but d'assurer mon équilibre et mon épanouissement tant personnel qu'académique. Merci d'avoir été mes premiers et inconditionnels supporteurs, de m'avoir assistée dans toutes mes entreprises, de m'avoir toujours encouragée à aller de l'avant et à donner le meilleur de moi-même. Merci d'avoir été mes modèles de vie : vos exemples de courage et de persévérance, face aux aléas de la vie, m'ont fortifiée et m'ont poussée à ne jamais baisser les bras lorsque des obstacles, si nombreux par moment, se dressent sur mon chemin. Merci pour votre dévouement, votre confiance et votre soutien financier jusqu'à ce jour. Et enfin, un grand merci pour votre support moral et votre amour qui n'ont jamais failli, et ceci, malgré la distance qui me sépare de vous : grâce à vos gestes d'affection et à vos mots d'encouragement, j'ai pu mieux appréhender ce long parcours et vivre cette expérience doctorale, un jour à la fois. MERCI, MERCI POUR TOUT !

RÉSUMÉ

Depuis une dizaine d'années, la dématérialisation de l'information connaît un essor particulier avec l'ascension du Cloud Computing. La puissance de calcul et de stockage offerte, ainsi que la flexibilité d'utilisation ont favorisé une externalisation accrue des données vers des serveurs distants. Ces derniers affranchissent les utilisateurs du fardeau de l'outil informatique traditionnel et leur donnent accès à un large éventail de services en ligne, qui sont des charges modulables, facturées selon l'utilisation. Particulièrement, dans le cas du modèle d'Infrastructure-service, le client dispose d'une infrastructure physique hébergée et peut ainsi louer des serveurs physiques, sur lesquels tourneront ses applications encapsulées dans des machines virtuelles ou Virtual Machines (VMs).

Toutefois l'émergence du Cloud et son adoption à grande échelle constituent des défis pour les fournisseurs d'infrastructure. En effet, au-delà de l'implantation et de la configuration des réseaux physiques, il faut conjuguer avec l'infrastructure sous-jacente déjà existante, et déterminer des mécanismes efficaces d'assignation des requêtes des usagers aux serveurs et data centers, contraint par le respect des performances des applications hébergées et des exigences de sécurité imposées par les clients. La demande sans cesse croissante et le souci de fournir une certaine qualité de service, obligent les fournisseurs à investir d'importants capitaux afin de multiplier leurs offres d'hébergement dans plusieurs zones géographiques. Avec ce déploiement à grande échelle d'énormes data centers, leur utilisation à outrance, l'augmentation des coûts d'opération et de l'énergie électrique, les dépenses d'exploitation ont rapidement dépassé les investissements. De ce fait, plusieurs auteurs se sont penchés sur le problème de placement des charges dans un environnement Cloud et ont développé des outils d'aide aux prises de décision, basés concomitamment sur l'accroissement des profits, une meilleure correspondance entre les besoins essentiels du client et l'infrastructure disponible, et la maximisation de l'efficacité et de l'utilisation des ressources.

Bien que le Cloud Computing offre une réponse favorable au problème de calcul et de stockage des informations, son adoption à grande échelle est freinée par les inquiétudes soulevées par sa signature écologique. En effet, l'utilisation excessive des data centers, de même que leur gestion et leur entretien, requièrent une énergie électrique accrue se traduisant par une empreinte carbone de plus en plus importante, accélérant ainsi le réchauffement climatique. Ainsi, au moment d'opter pour une solution Cloud, certains usagers questionnent l'impact environnemental d'un tel choix. Dans ce contexte, afin de favoriser l'expansion du Cloud, les géants de l'informatique n'ont d'autre choix que de conférer une dimension "vert" à leur

infrastructure physique. Ceci se traduit par des techniques d'assignation des charges visant à réduire l'empreinte carbone des data centers afin de faire du Cloud Computing un succès tant technologique qu'écologique.

Plusieurs études portant sur la réduction de l'empreinte carbone d'un unique data center, ont été récemment effectuées en considérant les techniques d'optimisation de l'énergie consommée. Toutefois, dans le contexte d'un Intercloud, où différents data centers sont géographiquement distribués et alimentés par des sources d'énergie renouvelables ou non, la consommation énergétique totale ne saurait refléter l'empreinte carbone dudit environnement. En ce sens, des recherches plus poussées ont porté sur l'optimisation de l'impact environnemental d'un InterCloud où l'hétérogénéité de l'infrastructure a été prise en compte. Cependant, seul le processus de placement des charges a été optimisé sans aucune considération pour l'amélioration de l'efficacité énergétique des data centers, pour la réduction de la consommation des ressources réseau, ou encore pour les exigences des clients en matière de performance des applications et de sécurité des données.

À cet effet, cette thèse propose un cadre de planification des applications visant à minimiser l'empreinte carbone dans un environnement InterCloud. Généralement, le problème est traité de manière globale, en combinant le choix de l'emplacement des applications et le routage du trafic associé au problème de gestion du système de refroidissement dans les différents data centers. Divers aspects, comme la puissance des équipements de calcul, la consommation des ressources réseau et l'efficacité énergétique seront simultanément optimisés, sous la contrainte des exigences des clients. Le travail a été réalisé en trois phases.

Dans le premier volet du travail, un modèle d'optimisation du placement des applications simples, à VM unique, a été développé afin de réduire l'impact écologique d'un ensemble de data centers. Le modèle d'empreinte carbone proposé améliore les approches de consommation énergétique déjà existantes en combinant l'optimisation du placement des VMs au mécanisme d'accroissement de l'efficacité énergétique des data centers. Ce dernier processus consiste à déterminer, pour chaque data center actif, la valeur optimale de température à fournir par le système de refroidissement, de manière à trouver un compromis entre les gains énergétiques, associés au cooling, et l'augmentation de la puissance des ventilateurs des serveurs, face à un accroissement de la température ambiante. Afin d'ajouter un certain réalisme au modèle, les exigences des clients, en termes de performances des applications hébergées, ou encore en rapport avec les notions de sécurité et de redondance, ont également été considérées. Une analyse de la monotonie et de la convexité du modèle non linéaire résultant a été effectuée afin de souligner l'importance entourant la détermination d'une valeur optimale de température. Par la suite, le problème a été transformé en un modèle linéaire et résolu de

manière optimale avec un solveur mathématique, à l'aide des techniques de programmation linéaire en nombres entiers. Afin de mettre en évidence la pertinence du modèle d'optimisation proposé en termes de coût d'empreinte carbone, une analyse de la structure du coût et de l'impact de la charge a été réalisée. Dans le but de mieux apprécier les résultats, une version simplifiée du modèle, exempte de toute exigence du client, a alors été considérée. Ce même modèle simplifié a également été comparé à différentes techniques visant à optimiser l'empreinte carbone autant au sein d'un unique data center qu'à l'échelle d'un environnement InterCloud. Les résultats ont démontré que le modèle proposé permet de réduire jusqu'à 65% le coût d'empreinte carbone. De plus, afin de souligner l'efficacité du modèle proposé à réaliser un placement des VMs tout en respectant les contraintes de sécurité et de performances, le modèle simplifié a été comparé au modèle intégrant les exigences des clients. Bien que le modèle sans contraintes génère, en général des coûts d'empreinte carbone inférieurs à celui du modèle complet, il demeure moins intéressant à considérer, car le gain d'empreinte carbone résultant du processus de consolidation aveugle ne permet pas de contrebalancer le pourcentage de violation des contraintes. Ces résultats ont également permis de démontrer les bonnes performances du modèle complet comparé à sa variante simplifiée, dans le sens que le premier permet parfois d'obtenir des configurations de coût identique au modèle simplifié, en s'assurant, de surcroît, du respect des exigences des utilisateurs.

Le mécanisme de placement des VMs est un problème complexe à résoudre. En raison de la nature NP-complet du problème, le temps de calcul croît de manière exponentielle en fonction des entrées et seules les instances de petite taille, même dans le cas du modèle simplifié, ont pu être résolues avec la méthode exacte. Afin de pallier ce problème, nous proposons, dans la seconde étape de notre travail, une méthode de résolution, basée sur les métaheuristiques, dans le but d'obtenir des solutions de qualité en un temps polynomial pour des instances de grande taille. La méthode de résolution proposée dans cet article est basée sur l'heuristique de Recherche Locale Itérée ou Iterated Local Search (ILS), qui implémente une descente comme mécanisme de recherche locale et, suite à l'arrêt prématuré du processus de descente, effectue des sauts dans l'espace des solutions afin de relancer l'exploration à partir d'une nouvelle configuration. Aussi, afin d'accélérer le processus d'évaluation d'une configuration, des fonctions de gains, traduisant la différence entre le coût de la solution actuelle et celui du voisin considéré, ont été déterminées. Divers mécanismes de perturbations ont également été implémentés afin d'éviter le piège des optima locaux. De manière générale, les résultats présentés sont de deux types : la paramétrisation de la méthode et l'évaluation des performances de l'algorithme. La phase de paramétrisation a permis de déterminer les mécanismes à implémenter à chaque étape de l'algorithme ainsi que la valeur idéale des paramètres clés. Par la suite, les performances de l'algorithme ont d'abord été comparées à

celles obtenues avec la méthode exacte définie au premier volet. Les résultats démontrent que les solutions générées par la méthode proposée sont en moyenne à environ 0.2% de la solution optimale, avec un écart maximal de 2.6% et un temps d'exécution moyen inférieur à 3 secondes. Afin d'analyser les performances générales de la méthode proposée, cette dernière a été exécutée sur différentes tailles d'instances du modèle et les résultats obtenus ont été évalués par rapport à ceux découlant de l'implémentation de trois méthodes approchées retrouvées dans la littérature. Les résultats ont pu démontrer que l'heuristique proposée permet d'établir un bon compromis entre la qualité de la solution et le temps d'exécution, et peut engendrer une économie de coût de carbone pouvant s'élever jusqu'à 34%.

Par ailleurs, des applications de plus en plus complexes, s'étendant sur plusieurs VMs, se font de plus en plus héberger dans le Cloud. Elles introduisent des trafics inter-VMs, sollicitant ainsi les ressources réseau afin de faire transiter l'information d'une machine virtuelle ou Virtual Machine (VM) à une autre. Or, comme la consommation énergétique des ressources réseau représente environ le quart de la puissance totale d'un data center, il en résulte alors que l'impact énergétique de ces équipements ne saurait être négligé plus longtemps, lorsque vient le temps de décider de l'emplacement des VMs afin de réduire l'empreinte écologique de plusieurs data centers. Dans ce contexte, la dernière phase de notre travail propose une extension du modèle développé à la première étape, où l'optimisation du placement des VMs est combinée au mécanisme d'amélioration de l'efficacité énergétique et au routage du trafic. De plus, le processus de routage du trafic étant également NP-complet, la combinaison de ce dernier au mécanisme de placement des VMs résulte en un problème encore plus difficile. En ce sens, nous avons également proposé une approche de résolution basée sur la combinaison de deux métaheuristiques, soit la Recherche Locale Itérée (ILS) et la Recherche Tabou (Tabu Search (TS)). De manière générale, la méthode développée implémente l'algorithme ILS où le mécanisme de recherche locale est une adaptation de l'heuristique TS. Cette hybridation permet de tirer profit des avantages des deux méthodes : d'une part, des mécanismes de mémoire à court et long terme afin d'éviter les cycles et les optima locaux, et d'autre part, des opérateurs de perturbation qui relancent l'exploration à partir d'une nouvelle configuration de départ. Dans la phase d'expérimentation, autant le modèle global que la méthode de résolution proposés ont été évalués. Le modèle global a été implémenté en AMPL/CPLEX en utilisant les techniques de programmation linéaire en nombres entiers et a été évalué par rapport à d'autres modèles de référence optimisant un unique objectif à la fois. Comme nous nous y attendions, le modèle proposé permet d'obtenir de meilleures configurations en termes de coût d'empreinte carbone, avec un gain pouvant s'élever jusqu'à environ 900% pour les instances considérées. Toutefois, cette optimisation se fait au prix d'un temps de calcul, en moyenne, relativement plus élevé, en raison de la complexité du modèle proposé. Cependant,

l'économie en carbone réalisée étant substantiellement plus importante comparée aux écarts en temps de calcul observés, les résultats ont pu démontrer la grande efficacité du modèle proposé par rapport aux modèles réalisant une optimisation mono-objectif. La méthode de résolution approchée proposée a été implémentée en C++ et des expériences préliminaires nous ont permis de dégager les valeurs optimales des paramètres clés de la méthode, dont la plupart sont liées à la taille du problème. L'efficacité de la méthode, en termes de compromis entre coût d'empreinte carbone et temps d'exécution, a d'abord été comparée par rapport aux valeurs de borne inférieure, pour les instances de petite taille. Les résultats montrent que l'algorithme développé est en mesure de trouver des solutions, en moyenne, à moins de 3% de la borne inférieure en un temps polynomial, contrairement à une croissance exponentielle du temps de calcul pour la méthode exacte. Pour les plus grandes instances, une comparaison avec différentes méthodes de référence a pu démontrer que l'approche proposée est toujours en mesure de trouver les configurations de coût minimal en un temps réduit, soulignant ainsi les bonnes performances de l'heuristique développée et la justesse au niveau du choix des paramètres de simulation qui y sont associés.

Ces expériences ont démontré, au regard des résultats obtenus, que le travail réalisé permettra d'offrir, aux fournisseurs de Cloud, des outils efficaces de planification des applications à l'échelle de leurs data centers, afin de mieux faire face aux inquiétudes soulevées quant à l'impact écologique du Cloud Computing sur le bien-être de la planète.

ABSTRACT

The last decade or so has seen a rapid rise in cloud computing usage, which has led to the dematerialization of data centers. The higher computing power and storage, combined with a greater usage flexibility, have promoted the outsourcing of data to remote servers, allowing users to overcome the burden of traditional IT tools, while having access to a wider range of online services that are charged on based on usage. Particularly, in the case of an infrastructure service model, the client is given access to a physical infrastructure where he can rent physical servers to run their applications, which are encapsulated in VMs.

However, the emergence of cloud service and its wide adoption impose new challenges on infrastructure providers. They have to optimize the underlying existing infrastructure by identifying efficient mechanisms for assigning user requests to servers and data centers, while satisfying performance and security constraints, as imposed by the clients. Faced with an increasing demand, providers have to invest significant capital in order to increase their hosting offers in several geographic areas, to provide the required Quality of Service (QoS). The increased use of data centers also has a huge bearing on the operating costs and energy consumption, as operating expenses have quickly exceeded the investment. Therefore, several authors have been tackling the placement of loads in a cloud environment by developing tools to aid in the decision-making. Most of the proposed solutions are guided by financial aspects to increase profits by determining the best mapping between the basic needs of the client and the available infrastructure, in order to meet the QoS constraints while maximizing the efficiency and the use of resources.

However, while cloud computing represents a great opportunity for both individuals and businesses, its widespread adoption is slowed by concerns regarding its global ecological impact. The excessive use of data centers, as well as their management and maintenance, require huge amounts of electric power, thus accelerating the global warming process. Therefore, before choosing a cloud solution, some users will take into consideration that environmental impact. This, in turn, forces cloud providers to consider the "green" aspect of their infrastructure by developing new ways of assigning loads that reduce the carbon footprint of their data centers in order for cloud computing to be both a technological and an ecological success.

Several works have recently been published that aim to reduce the environmental impact of clouds. The first ones have focused on reducing the carbon footprint of a single data center by optimizing the consumed energy. However, in the context of an InterCloud environment composed of different data centers that are geographically distributed and powered by re-

newable energy sources or not, the total energy consumed cannot reflect the carbon footprint of that said environment. That's why subsequent research has been focusing on optimizing the environmental impact of an InterCloud where the heterogeneity of the infrastructure is taken into account. However, only the VM placement process has been optimized with no consideration to improving data center energy efficiency, network power consumption or customer requirements, as far as application performance and data security are concerned.

To this end, this thesis proposes a framework for assigning applications to an InterCloud with the view of minimizing the carbon footprint of such a computing environment. In order to address this issue, the problem is treated holistically, jointly optimizing the VM placement process, the traffic routing and a cooling management technique that considers the dynamic behavior of the IT fans. Various aspects, such the processing power, the network resource consumption and the energy efficiency, will be simultaneously optimized, while satisfying customer requirements. The work is carried out in three phases.

First, we propose an optimization model for placing standalone VMs, in order to reduce the environmental impact of a set of data centers. The carbon footprint of the proposed model improves the energy consumption of existing approaches by combining both the optimization of the VM placement process and the energy efficiency of data centers. This latter process determines, for each active data center, the optimal temperature to be provided by the cooling system so as to find a compromise between the energy gains associated with the cooling and the increased power consumption of the server fans, at high temperatures. In order to add some realism to the model, customer requirements are also considered, in terms of application performance, security and redundancy. An analysis of the monotony and the convexity of the resulting nonlinear model was conducted to highlight the importance surrounding the determination of the optimal temperature value. Subsequently, the problem is transformed into a linear model and solved optimally with a mathematical solver, using integer linear programming. To demonstrate the relevance of the proposed optimization model in terms of carbon footprint, an analysis of the cost structure and the impact of the load is carried out. In order to better highlight the results, a simplified version of the model, free from any client requirements, is also considered. This same simplified model is also compared with different other techniques that optimize the carbon footprint both within a single data center and across an InterCloud environment. The results demonstrate that the proposed model can yield savings of up to 65%, in terms of carbon footprint cost. In addition, to highlight the effectiveness of the proposed model when placing VMs while satisfying clients and performance constraints, the simplified model is compared with the global model that incorporates customer requirements. Although the model without constraints usually generates smaller carbon footprint costs, its result is not as interesting as it may seem, because

these savings do not offset the cost of violating constraints. These results also demonstrate the good performance of the global model compared to its simplified variant, in the sense that the first sometimes provides configurations identical to the simplified cost model, while ensuring that user requirements are met.

Placing VMs is a complex problem to solve. Due to its NP-complete nature, the computing time grows exponentially in the length of the inputs. Even with the simplified model, only small instances can be solved with the exact method. In order to overcome this problem, we propose, in the second stage of our work, a resolution method based on metaheuristics in order to obtain good solutions for large instances in a polynomial time. The method proposed in this article is based on the ILS heuristic that implements a descent as a local search mechanism and, following the early termination of the descent, performs jumps in the solution space to restart the algorithm from a new configuration. Furthermore, in order to speed up the evaluation of a configuration, gain functions that reflect the cost difference between the current solution and the considered neighbor, are determined. Various perturbation mechanisms are also implemented to avoid the trap of local optima. In general, the presented results are of two types: the parametrization of the method and the performance evaluation of the proposed algorithm. The parametrization phase helps determine the mechanisms to implement for each step of the algorithm as well as the ideal value of each key parameter. Thereafter, the algorithm performances are first compared with those obtained with the exact method defined in the first phase. The results demonstrate that the solutions generated by the proposed method are on average about 0.2% of the optimal solution, with a maximum deviation of 2.6% and an average running time of less than three seconds. To analyze the overall performance of the proposed method, the latter is run on instances of different sizes and the obtained results are compared with those resulting from the implementation of three resolution methods found in the literature. The results demonstrate that the proposed heuristic establishes a good compromise between the quality of the solution and execution time, with a carbon cost savings of up to 34%.

Moreover, complex applications spanning multiple VMs are increasingly hosted in the cloud and introduce inter-VM traffic and network resources seeking to transit information from one VM to another. However, as the energy consumption of network resources represents about 25% of the total power of a data center, its impact cannot be ignored any longer, when we decide on location of VMs to reduce the environmental footprint of multiple data centers. In this context, the last phase of our work proposes an extension of the model developed in the first stage, where the optimization of the placement of VMs is combined with the mechanism of improving energy efficiency and the traffic routing. In addition, the process of routing traffic is also NP-complete and combining it with the problem of placing the VMs results

in an even more difficult problem. Therefore, we also propose a resolution approach based on the hybridation of two metaheuristics, namely ILS and TS. In general, the developed method implements the ILS algorithm where the local search mechanism is an adaptation of the TS heuristic. This hybrid approach leverages the advantages of both methods, namely short and long term memory mechanisms in order to avoid cycles and local optima, as well as perturbation operators that boost exploration from a new starting configuration. In the experimental phase, both the proposed global model and resolution method are evaluated. The model is implemented in AMPL/CPLEX using integer linear programming and is evaluated by comparing it with other reference models with one goal being optimized at a time. As expected, the proposed model allows for better configurations in terms of carbon footprint cost, with a gain of up to 900% for the considered instances. However, this optimization is done at the cost of a computing time that is relatively higher because of the complexity of the proposed model. However, the carbon savings being substantially greater than the computing time differences, the results show the great efficiency of the proposed model when compared with models performing a single-objective optimization. The proposed resolution method has been implemented in C++ and preliminary experiments allow us to identify the optimal values of key parameters of that method, most of which depend on the problem size. The effectiveness of the method in terms of compromise between cost and carbon footprint runtime is first compared against the lower limit values of small instances. The results show that the developed algorithm is able to find solutions that are on average within less than 3% of the lower bound, and are computed in polynomial time, unlike exponential growth of the exact method. For larger instances, a comparison with different reference approaches demonstrates that the proposed approach is always able to find the minimum cost configurations in less time, highlighting the good performance of the proposed heuristic and accuracy in the choice of simulation parameters associated with it.

All these experiments demonstrate, in light of the results obtained, that our work will provide to cloud suppliers effective tools for planning their applications across their data centers in order to better address the concerns regarding the environmental impact of Cloud Computing on the well-being of the planet.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

AC	Air Conditioning
ASHRAE	American Society of Heating, Refrigerating and Air Conditioning Engineers
BF	Best Fit
BFD	Best Fit Decreasing
BPP	Bin-Packing Problem
CED	Carbon Emission Directory
CEGP	Carbon Efficient Green Policy
CF	Consolidation Fitness
COP	Coefficient of Performance
CRAC	Computer Room Air Conditioner
DVFS	Dynamic Voltage Frequency Scaling
EDF	Earliest Deadline First
FFD	First Fit Decreasing
FTSFTW	Follow The Sun/Follow The Wind
GES	Gaz à Effet de Serre
GF	Greenness factor
GhG	Greenhouse Gas
GLB	Geographical Load Balancing
GOD	Green Offer Directory
IaaS	Infrastructure as a Service
ICT	Information and Communication Technology
IETF	Internet Engineering Task Force
ILP	Integer Linear Programming
ILS	Iterated Local Search
IP	Integer Programming
ITS	Iterated Tabu Search
LARIM	LAboratoire de recherche en Réseautique et Informatique Mobile
LS	Local Search
MBFD	Modified Best Fit Decreasing
MILP	Mixed-Integer Linear Programming
MINLP	Mixed-Integer Nonlinear Programming
MLGGA	Multi-Level Grouping Genetic Algorithm

OS	Operating Systems
OSI	Open Systems Interconnection
OVMP	Optimal Virtual Machine Placement
PaaS	Platform as a Service
PUE	Power Usage Effectiveness
QdE	Qualité de l'Expérience
QdS	Qualité de Service
QoE	Quality of Experience
QoS	Quality of Service
RL	Resource Limit
SaaS	Software as a Service
SA	Simulated Annealing
SLA	Service Level Agreement
TIC	Technologies de l'Information et de la Communication
TS	Tabu Search
VM	Virtual Machine
VMPP	Virtual Machine Placement Problem
VMs	Virtual Machines

CHAPITRE 1 INTRODUCTION

Depuis quelques années, le monde des télécommunications assiste à l'émergence d'un nouveau paradigme, le Cloud Computing, qui permet aux utilisateurs d'externaliser leurs applications et d'exploiter des ressources informatiques physiques à travers Internet, sans avoir à gérer l'infrastructure sous-jacente, souvent complexe (Armbrust *et al.*, 2010). La disponibilité d'une infrastructure informatique hébergée, où l'accès aux ressources est sans contraintes, devient alors primordiale pour l'usager (Anderson *et al.*, 2005). Ce nouveau modèle de service introduit donc de nouveaux défis dans l'univers de la planification où il s'agit de définir des mécanismes de mise en correspondance entre les requêtes des usagers et les ressources du support physique. Dans un tel contexte, les fournisseurs se doivent de satisfaire les besoins des clients, en offrant des services à moindres coûts tout en s'assurant du respect des exigences de ces derniers, sous le couvert d'une utilisation efficace des ressources physiques. Aussi, afin de faire main basse sur une grande part du marché et accroître leur profit, certains fournisseurs vont même jusqu'à déployer des data centers à des endroits différents, exposant ainsi leurs offres de services à une plus grande clientèle.

Jusqu'à récemment, la planification dans le Cloud s'était penchée sur l'amélioration de métriques telles que la performance des applications, la disponibilité du service et la fiabilité du réseau (Dupont *et al.*, 2012). Toutefois, en raison des nombreux avantages du Cloud, ce dernier a rapidement gagné en popularité. La demande sans cesse croissante a alors mené à la création de data centers à grande échelle, dont la gestion requiert de grandes quantités d'énergie électrique (Mazzucco *et al.*, 2010). En raison du prix de l'énergie qui monte en flèche et des coûts d'opération dépassant largement les dépenses d'investissement, l'efficacité énergétique des data centers a rapidement commencé à susciter un vif intérêt. Une planification rigoureuse permettant de satisfaire la demande croissante, tout en réduisant les coûts d'exploitation, est donc requise afin de permettre aux fournisseurs de demeurer compétitifs.

Cependant, au problème d'accroissement énergétique se rattache un souci beaucoup plus inquiétant qu'une simple réduction de la marge de profit des propriétaires de Cloud : avec une empreinte carbone représentant actuellement environ 2% des émissions de CO_2 dans l'air, l'usage excessif des infrastructures Cloud contribue largement au problème du réchauffement climatique. Aussi, pour freiner l'impact néfaste et sans cesse grandissant du Cloud sur la santé planétaire, éviter de violer les limites d'émission de Gaz à Effet de Serre (GES) qu'imposent les organismes gouvernementaux à l'industrie des Technologies de l'Information et de la Communication (TIC)s, et favoriser l'adoption à grande échelle des technologies Cloud, les

géants de l'informatique se doivent de conférer une dimension "vert" à leur infrastructure. L'enjeu associé à la planification dans le Cloud n'est plus uniquement économique mais est intimement lié à la notoriété et à l'image de ces entreprises Cloud. Ainsi, pour mieux faire face à la rude concurrence, les fournisseurs n'ont pas d'autres choix que de miser sur des techniques de planification visant à réduire l'impact écologique de leurs data centers.

Dans cette thèse, nous nous intéressons à la minimisation de l'empreinte carbone dans un environnement InterCloud, où le problème est traité de manière globale, combinant la détermination de l'emplacement des applications, incluant le routage du trafic, au problème de gestion du système de refroidissement dans les différents data centers. Plusieurs paramètres, à savoir la puissance des nœuds de calcul, la consommation des ressources réseau et l'efficacité énergétique, seront simultanément optimisés, sous la contrainte des exigences des clients.

Ce chapitre d'introduction est donc divisé comme suit. Dans un premier temps, les principaux concepts et définitions permettant de mieux cerner le cadre du travail seront présentés. Par la suite, les éléments liés à la problématique visant à réduire l'empreinte carbone seront exposés, et seront suivis par les objectifs de recherche. Les principales contributions faisant l'originalité du travail de recherche seront ensuite mises en évidence, et une esquisse des chapitres subséquents constituant cette thèse permettra de clore cette section.

1.1 Définitions et concepts de base

Cette section vise à présenter au lecteur les concepts et définitions permettant de situer le cadre du travail et qui seront utilisés tout au long de ce document. Nous commencerons par introduire le Cloud Computing et ses caractéristiques, suivi des notions rattachées au processus d'hébergement des applications et des concepts liés aux exigences des clients. Par la suite, les termes clés relatifs à la puissance consommée [1] dans un Cloud, ainsi qu'à l'empreinte carbone générée seront expliqués. Nous terminerons cette section en présentant les principales étapes inhérentes au processus d'hébergement des applications dans le Cloud.

1.1.1 Cloud Computing

Avec le monde de l'informatique qui devient de plus en plus vaste et complexe, le Cloud Computing s'érige comme un nouveau modèle permettant de répondre aux besoins sans cesse croissants des utilisateurs. Plus spécifiquement, il fait référence à l'utilisation de mémoires et de capacités de plusieurs ordinateurs répartis dans le monde et reliés par Internet, dans le but de faciliter le stockage et le traitement d'un grand volume de données. Les applications

1. Puissance et énergie seront utilisées de manière interchangeable.

et les données qui, jadis, se trouvaient sur les ordinateurs personnels, migrent vers d'énormes centres de traitement composés de plusieurs serveurs interconnectés mais distants. Les utilisateurs peuvent ainsi accéder à des services en ligne, sans être propriétaires de leurs serveurs informatiques, et être facturés selon leur utilisation. Ceci leur permet de grandes économies au niveau des supports et leur évite la gestion, souvent complexe, de l'infrastructure physique sous-jacente (Fox *et al.*, 2009). De manière générale, le Cloud Computing se caractérise par différents modèles de service et 4 modèles de déploiement.

1.1.1.1 Modèles de service

Le Cloud Computing fait intervenir généralement trois modèles de service présentés ci-dessous et illustrés à la Figure 1.1 (GRANGE, 2010) :

Logiciel-service Le logiciel-service ou *Software as a Service (SaaS)* est un modèle économique où le client utilise les applications du fournisseur, lesquelles sont hébergées sur le Cloud. Google Apps, Google Docs, Office Web Apps, LotusLive sont les principaux logiciels-services ayant émergé de ce nouveau paradigme (voir Anderson *et al.*, 2005).

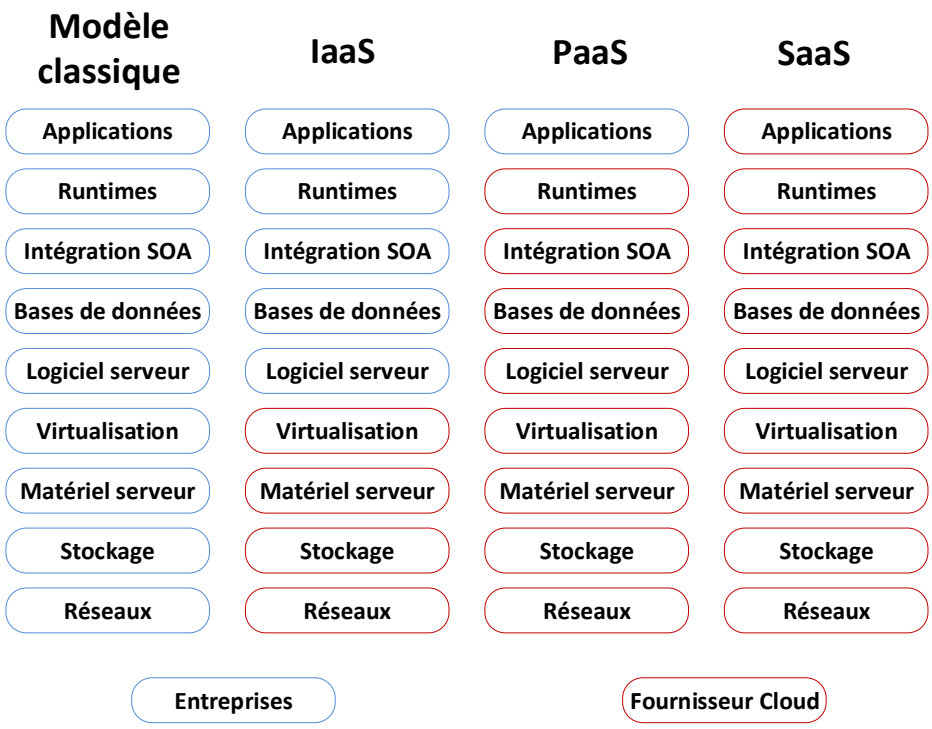


Figure 1.1 Modèles de service

Plateforme-service La plateforme-service ou *Platform as a Service (PaaS)* fournit des environnements de création et d'exécution pour des applications créées en utilisant des langages et des outils supportés par le fournisseur. Ainsi, le consommateur développe ses propres applications qui sont stockées sur le Cloud. Parmi ces plateformes-services, on distingue Google App Engine de Google, Azure d'Amazone et Force.com de SalesForces (voir Peng *et al.*, 2009).

Infrastructure-service L'infrastructure-service ou *Infrastructure as a Service (IaaS)* est utilisé pour la mutualisation des infrastructures entre les clients et les entreprises. Dans ce modèle, le client contrôle les applications, le logiciel serveur, les bases de données, les systèmes virtuels, alors que le fournisseur gère la virtualisation, le matériel serveur, le stockage et les infrastructures réseau. L'« Elastic Cloud Compute (EC2)» d'Amazone est un exemple d'IaaS (Amazon, 2010).

1.1.1.2 Modèles de déploiement

Le Cloud peut être déployé de plusieurs manières distinctes, comme l'illustre la Figure 1.2 :

Cloud privé L'infrastructure est uniquement exploitée par une organisation privée, cependant la gestion de ce nuage peut revenir à une tierce partie.

Cloud public L'infrastructure est mise à la disposition du public en général ou d'une organisation très large. Toutefois, ce Cloud appartient à un fournisseur de services.

Cloud communautaire L'infrastructure est utilisée par plusieurs organisations adhérant à une cause précise et partageant les mêmes idéologies. Ce Cloud peut être géré par la communauté ou par un tiers.

Cloud hybride Il est composé de plusieurs nuages demeurant des entités à part entière, mais qui sont reliés, par le biais de technologies standards ou propriétaires, dans le but de faciliter la portabilité des applications et des données.

1.1.2 Caractéristiques techniques du Cloud Computing

Cette section présente les éléments constitutifs du Cloud, et plus précisément ses caractéristiques techniques.

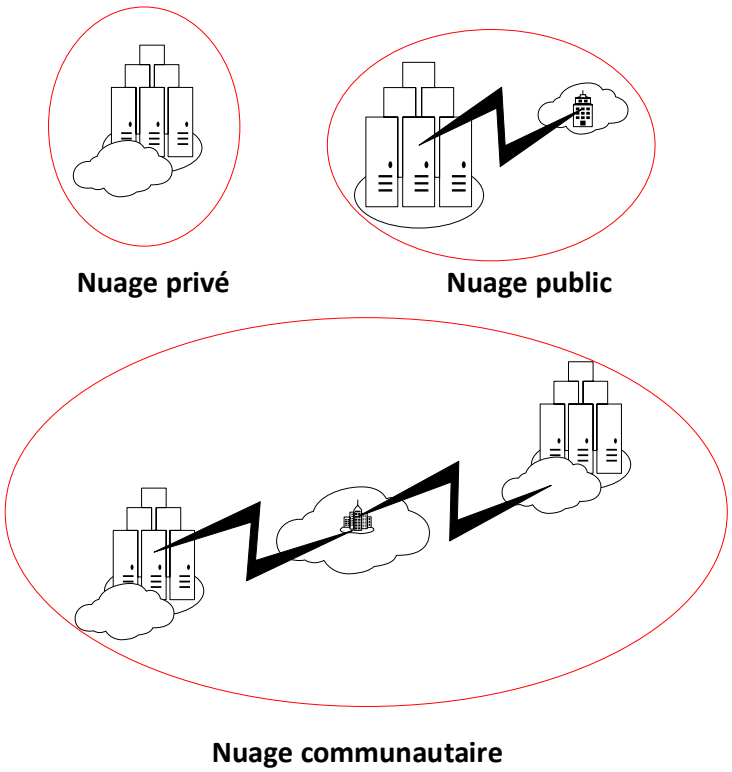


Figure 1.2 Modèles de déploiement

1.1.2.1 Data center

Dans le paradigme du Cloud Computing, les applications sont hébergées dans un ou plusieurs data centers, lequel se définit, logiquement, comme un réseau de serveurs interconnectés entre eux, et se présente sous la forme d'une architecture à plusieurs niveaux : ainsi, on distingue les commutateurs d'accès directement connectés aux serveurs, le réseau d'agrégation et le réseau cœur (Meng *et al.*, 2010).

Topologies des data centers La Figure 1.3 illustre les principales topologies des data centers. L'architecture la plus populaire est le modèle en arbre ou *Tree*. On retrouve aussi le VL_2 où les commutateurs d'agrégation et ceux du réseau cœur forment un graphe biparti complet, permettant ainsi d'effectuer du load-balancing sur les liens. Un troisième exemple d'architecture est celui du *Fat-Tree* où les commutateurs d'accès et d'agrégation forment plusieurs graphes bipartis, des gousses, et où chaque gousse est reliée à un commutateur du réseau cœur, créant ainsi un second graphe biparti complet. La dernière topologie, *BCube*, intègre les serveurs au processus d'acheminement des données.

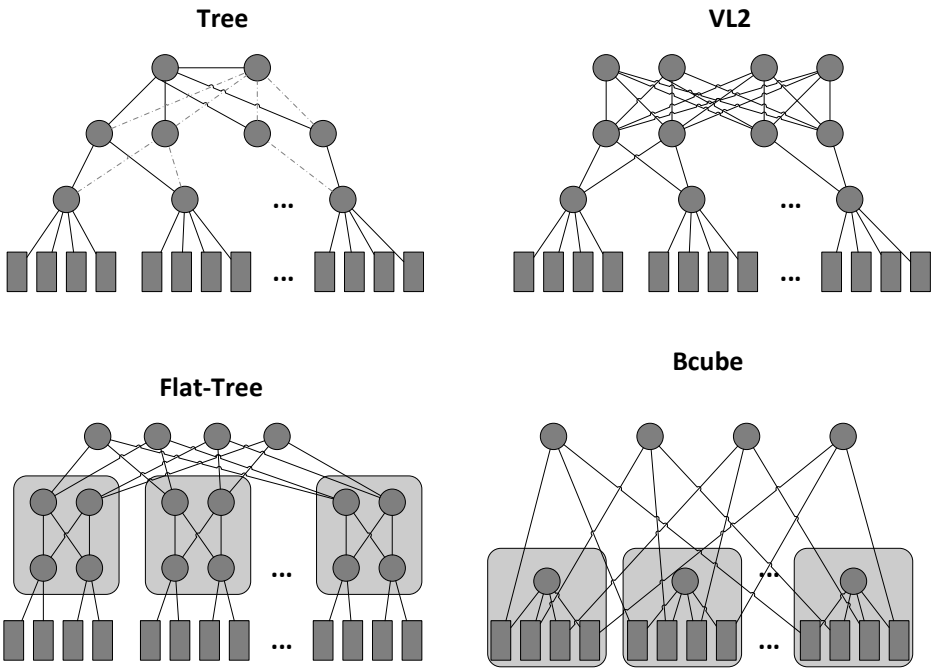


Figure 1.3 Topologies des data centers

Configuration interne d'un data center Physiquement, un data center est une installation abritant des rangées de serveurs, comme l'illustre la Figure 1.4. Chaque rangée est composée de racks de châssis, et chaque châssis regroupe plusieurs serveurs lame ou *Blade servers*, lesquels partagent les mêmes ventilateurs et sources d'alimentation. Un système de refroidissement ou *Computer Room Air Conditioner (CRAC)* est installé dans la pièce de telle sorte que les racks soient emprisonnés entre une allée chaude et une allée froide (Pakbaznia et Pedram, 2009). L'air frais des allées froides, fourni par le CRAC, pénètre la pièce en traversant les grilles installées au sol et est absorbé par les serveurs. Une fois refroidis, les serveurs rejettent l'air chaud qui est, par la suite, récupéré par le système de refroidissement. En l'absence de séparateurs physiques, une partie de l'air chaud s'échappant des serveurs peut être redistribuée dans la pièce. C'est le principe de la recirculation de la chaleur.

1.1.2.2 InterCloud

Afin d'atteindre un plus grand nombre de clients, pour réduire les coûts d'exploitation (Garg *et al.*, 2011a), ou encore, en vue de satisfaire les besoins des usagers en termes de délai minimal, certaines fournisseurs Cloud déploient des data centers dans plusieurs régions. Ce regroupement de data centers, géographiquement distribués, interconnectés entre eux et appartenant à un unique fournisseur est communément appelé InterCloud. La Figure 1.5 dépeint

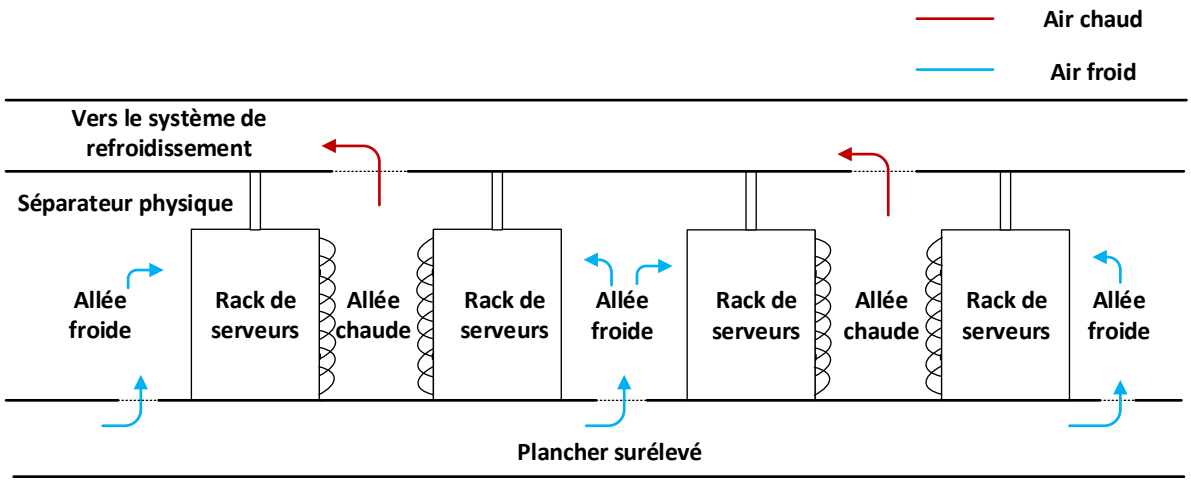


Figure 1.4 Configuration interne d'un data center

un InterCloud constitué de n data centers.

1.1.2.3 Virtualisation

La virtualisation se définit comme un ensemble de techniques permettant de regrouper sur un seul support physique, des ressources informatiques afin qu'elles puissent effectuer séparément des tâches spécifiques, comme si elles étaient exécutées sur des supports physiques différents (ESSEC, 2009). En ce qui a trait à l'implémentation, la virtualisation consiste à fournir un environnement logique indépendant du support physique qui voit « de plus haut » les ressources physiques et facilite une gestion efficace de ces dernières. Avec la virtualisation, émerge également la notion de machines virtuelles qui fera l'objet de notre prochaine section.

1.1.2.4 Machine virtuelle

Une machine virtuelle ou Virtual Machine (VM), voir la Figure 1.6, se définit comme un environnement d'exécution qui reproduit le comportement d'un système hôte. Grâce à la virtualisation, plusieurs logiciels, isolés les uns des autres, peuvent fonctionner simultanément sur un seul support physique. Une VM est uniquement composée de logiciels de telle sorte que le processeur, la mémoire RAM, le disque dur et les cartes d'interface réseau sont tous des composantes virtualisées.

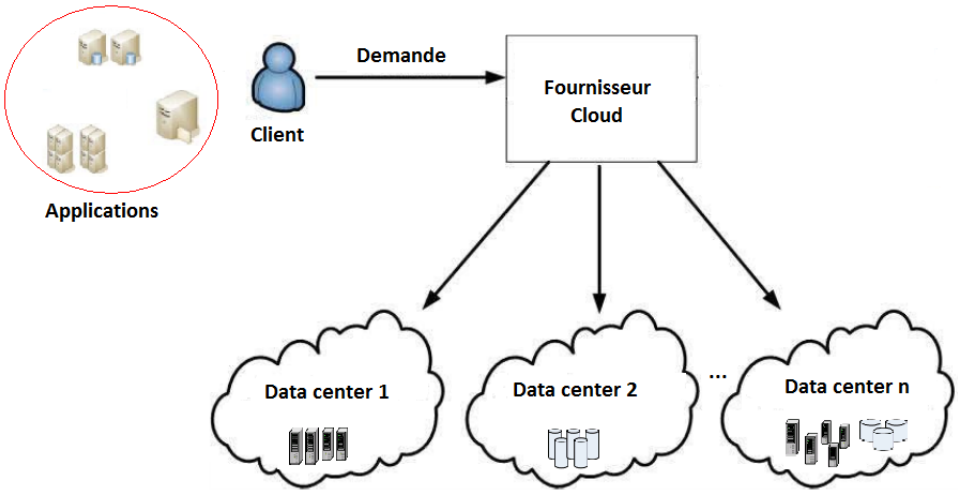


Figure 1.5 InterCloud

1.1.2.5 Trafic

Les applications hébergées sur le Cloud peuvent être de natures différentes, de l'application la plus simple, encapsulée dans une unique VM, aux applications complexes, tel un moteur de recherche, nécessitant des interactions entre plusieurs entités. Pour ce dernier cas, ces échanges introduisent un trafic non négligeable, lequel trafic se définit comme l'information circulant dans le réseau à un moment donné. Dans l'environnement Cloud, on distingue le trafic inter-VMs, où les données circulent à l'intérieur d'un data center ou entre data centers ; et le trafic externe, traduisant les échanges entre VMs et clients, comme dans le cas des téléchargements ou téléversements de données.

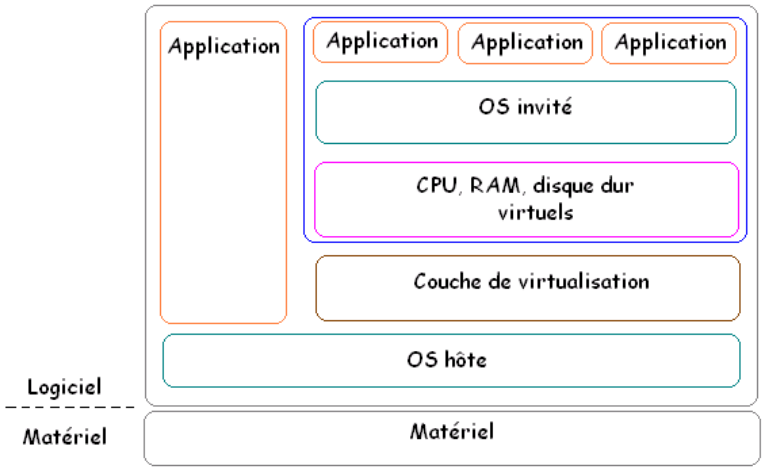


Figure 1.6 Machine virtuelle

1.1.3 Exigences du client

L'avènement du Cloud Computing ne fait pas que révolutionner le monde des technologies de l'information, mais il est à l'origine d'un changement profond au niveau du mode de travail des individus et apporte de nombreux avantages aux entreprises utilisatrices. Toutefois, afin de satisfaire les intérêts des clients, les fournisseurs se doivent de faire face à diverses exigences.

1.1.3.1 Qualité du service fourni

La concurrence féroce impose aux fournisseurs de Cloud d'offrir des services à moindres coûts tout en en assurant la bonne qualité. Cette dernière permet de déterminer si le service parvient à combler les attentes du client et se mesure différemment, selon que l'évaluateur soit le fournisseur de services ou l'utilisateur (Gozdecki *et al.*, 2003).

Qualité de Service À l'heure où on s'intéresse aux économies matérielles associées au Cloud, les enjeux relatifs aux performances des plateformes de prestation de services deviennent de plus en plus importants pour les fournisseurs : le développement d'une infrastructure réseau puissante devient un élément incontournable, et la conception du contenu des applications Web doit être adaptée aux navigateurs les plus populaires et les plus performants. À cet effet, les propriétaires de Cloud évaluent le degré de satisfaction des clients en se basant sur des paramètres techniques, définis comme des métriques de Qualité de Service (QdS). Ces mesures se rapportent aux performances de l'infrastructure physique sous-jacente, comme le débit, la gigue et la latence, sans nécessairement tenir compte de la perception des usagers, laquelle peut, de surcroît, différer d'un utilisateur à l'autre.

Qualité de l'Expérience La Qualité de l'Expérience (QdE), une autre famille de mesures liées à la QdS, et qui détermine le niveau d'appréciation de l'usager par rapport à un service offert, s'est progressivement imposée dans le monde des télécommunications. Elle vise à définir des métriques afin d'évaluer les facteurs influençant l'opinion de l'usager (Laghari *et al.*, 2011). Avec l'émergence du Cloud, la QdE devient un facteur important, particulièrement avec le modèle logiciel-service, où l'accès aux applications et aux données hébergées sur le Cloud doit se faire en temps réel. En ce sens, une mesure d'appréciation du service serait alors le délai perçu lors des tentatives d'accès aux informations en ligne. Alors qu'un délai faible entraînerait, de manière générale, une meilleure appréciation du client, la grande difficulté réside particulièrement dans l'évaluation de cedit faible délai, car ce niveau d'appréciation découle des attentes, perceptions et satisfactions d'un client particulier, à l'égard d'un service offert. En somme, les fournisseurs d'infrastructure Cloud se doivent également de considérer

la QdE bien que difficile à évaluer, car sujette à une très grande subjectivité, afin de satisfaire les intérêts des usagers, de s'assurer de leur fidélité et de la longévité des services offerts, et ainsi demeurer très compétitifs sur le marché (Gozdecki *et al.*, 2003).

Performance des applications hébergées Grâce aux techniques de virtualisation, plusieurs VMs peuvent être localisées sur le même serveur physique. Ceci peut, malheureusement, être à l'origine de conflits lorsque plusieurs applications compétitionnent pour accéder à des ressources physiques, limitant ainsi les performances des applications hébergées. Ainsi, considérant la nature des applications en question et les paramètres de qualité de service auxquels elles sont sensibles, il importe d'utiliser des techniques d'isolation ou de développer des mécanismes efficaces d'assignation des VMs aux serveurs afin de s'assurer que ces dernières soient déployées sur les serveurs hôtes adéquats.

Comme la QdE peut également se définir comme une QdS perçue par l'usager, et que la performance des applications hébergées dans le Cloud peut également se référer à la qualité du service offert, dans la suite du document, le terme QdS sera utilisé pour traduire simultanément la QdS intrinsèque définie par le fournisseur, la QdE déterminée par le client et le niveau de performance des applications hébergées sur le Cloud.

1.1.3.2 Enjeux de sécurité, de gouvernance et de redondance

L'externalisation des données soulève des inquiétudes à différents niveaux, notamment en matière de gestion et de sécurité des informations (A Vouk, 2008).

Sécurité Compte tenu de la valeur qu'ont les données dans notre civilisation immatérielle, une compagnie qui publie des documents confidentiels en ligne, doit se remettre à la sécurité du fournisseur, notamment pour ce qui est des cyber-attaques, mais aussi, par rapport aux données de ses compétiteurs qui peuvent se retrouver sur le même Cloud. De ce fait, il importera, pour les fournisseurs, d'implanter des mécanismes d'authentification des usagers et garantissant une connexion sécurisée. Aussi, en regard du nombre de serveurs, de la mutualisation et de la délocalisation de ceux-ci, il serait également intéressant de recourir au cryptage informatique dans le but d'assurer la confidentialité des données.

Gouvernance La gouvernance est un autre aspect de sécurité qui se réfère à la question de la territorialité des serveurs sur lesquels transitent ou sont stockées les données. En effet, les informations hébergées par un data center quelconque relevant souvent d'un régime juridique local, il est important, pour un utilisateur, de s'informer des politiques et des lois d'accès à

l'information propre à ce pays, mais également de s'assurer que ces règles s'appliquent dans son propre pays. Cependant, un point de complication persiste. En effet, à l'ère de l'économie globale et mondiale, il n'est pas impossible que les fournisseurs de services d'un pays riche et industrialisé utilisent des serveurs et des bases de données localisés dans les pays du Tiers-Monde. À cette phase, il importe, pour l'utilisateur, de déterminer les lois et les politiques de confidentialité qui prévalent et d'être assuré des moyens mis en place pour sauvegarder ses données ou les restaurer complètement dans des délais contractuels.

Redondance D'un autre côté, les risques encourus lorsque des données sont consolidées sur un unique serveur sont très élevés, car lors d'une panne informatique (*crash*), il peut s'avérer difficile de récupérer tous les renseignements, en l'absence de sauvegarde (*backup*). De ce fait, la réplication des informations sur plusieurs sites distants est un impératif de sécurité.

Bien que le Cloud Computing semble être une option très fiable, en raison de certaines déconvenues, la sécurité est perçue comme l'enjeu le plus critique autant du point de vue des fournisseurs que des usagers (Brunette *et al.*, 2009). Ainsi, des outils de cryptage des données et de gestion des identifiants peuvent être implantés, du côté fournisseur pour pallier le problème de sécurité. Les clients se doivent, également d'avoir un droit de regard en ce qui a trait à l'intégrité, à la disponibilité et à la confidentialité de leurs données, en spécifiant, par exemple des contraintes de localisation de leurs applications, afin d'éviter les compétiteurs, de gérer la question de la territorialité et d'assurer une certaine redondance.

1.1.4 Consommation énergétique dans le Cloud

La demande croissante en puissance de calcul et en espace de stockage a mené à la création de data centers à grande échelle. Or, bien que de telles infrastructures informatiques aient été mises en place afin de réduire l'importance de la puissance consommée, la gestion et le fonctionnement de ces énormes data centers nécessitent de grandes quantités d'énergie électrique (Fan *et al.*, 2007) ce qui, non seulement, entraîne des coûts d'opération largement supérieurs aux investissements, mais est également à l'origine d'émission de grandes quantités de GES et particulièrement de CO_2.

1.1.4.1 Répartition de la puissance consommée dans un data center

Comme l'illustre la Figure 1.7, les deux principales sources qui se partagent la facture énergétique, dans un data center typique, sont les équipements informatiques, qui représentent

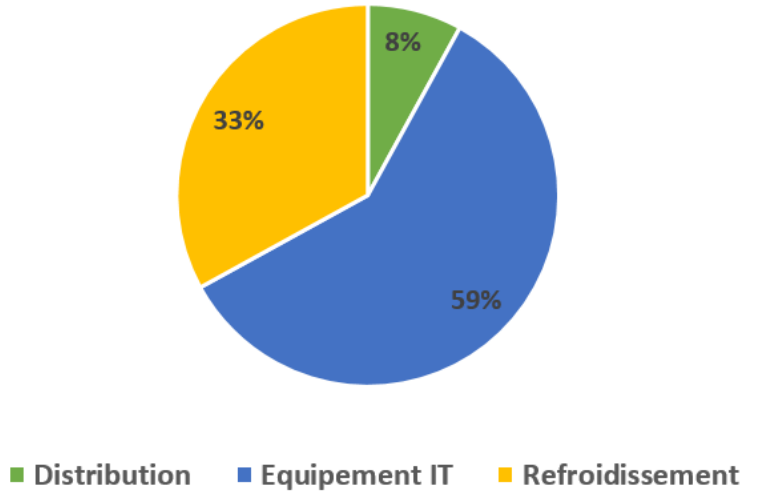


Figure 1.7 Répartition de la consommation de l'énergie dans un data center

environ 60% de la puissance consommée, et le système de refroidissement (cooling), comptant pour environ 30% de l'énergie totale. Le système de distribution, les lumières et autres équipements électriques se partagent les 10% restants. Les sections qui suivent décriront le mécanisme de transfert de chaleur, ainsi que le principe de consommation des équipements informatiques et du système de refroidissement.

1.1.4.2 Mécanismes de transfert de chaleur

L'équilibre thermodynamique à l'intérieur d'un data center dépend de plusieurs facteurs, notamment, de la température fournie par le système de refroidissement, de la puissance consommée par les équipements informatiques et de la configuration interne du data center. Dans le modèle serveurs lame, le châssis, regroupant des serveurs, une source d'alimentation et des ventilateurs, constitue l'élément de base de la configuration interne du data center (Pakbaznia et Pedram, 2009), ainsi, l'analyse du transfert de chaleur qui suit se fera à l'échelle des châssis. En considérant la loi de la conservation de l'énergie, la quantité de chaleur, Q_{out}, rejetée par un châssis dépend de la puissance consommée par ce dernier P, et de la quantité de chaleur d'entrée, Q_{in}, comme l'illustre l'équation suivante :

$$Q_{out} = Q_{in} + P \tag{1.1}$$

Dans le domaine des températures, on obtient de préférence :

$$T_{out} = T_{in} + P/R \tag{1.2}$$

où T_{in} et T_{out} représentent, respectivement, la température d'entrée et de sortie du châssis, et R se définit comme la résistance thermique du châssis, laquelle résistance dépend du flux d'air chaud sortant du châssis, de la densité de l'air et du coefficient de chaleur spécifique de l'air. De manière générale, la température d'entrée d'un châssis dépend de la température de l'air fourni par le système de refroidissement et de l'air chaud provenant de la recirculation de la chaleur s'échappant des châssis avoisinants. Toutefois, la présence de séparateurs physiques annule tout phénomène de recirculation, entraînant ainsi le fait que la température d'entrée d'un châssis soit égale à la température fournie par le système de refroidissement.

1.1.4.3 Consommation des équipements

Les équipements informatiques, grands consommateurs d'énergie, dans un data center, regroupent les nœuds de calcul et les commutateurs du réseau d'interconnexion. Dans cette thèse, le modèle de serveurs lame ou *blade servers* est utilisé, de ce fait, un nœud de calcul fait référence à un châssis regroupant un certain nombre de serveurs partageant les mêmes ventilateurs et sources d'alimentation.

Ventilateurs De manière générale, on recense trois (3) catégories de ventilateurs (Moss et Bean, 2009). La catégorie la plus connue est celle des ventilateurs à vitesse constante, où la puissance consommée est quasiment fixe ou varie très faiblement en fonction de la température d'entrée. La Figure 1.8 illustre le comportement d'un tel type de ventilateur. Toutefois, ce type de ventilateur peut se révéler inefficace dans plusieurs situations. En effet, à basse température, le besoin en refroidissement est faible, de ce fait, une vitesse fixe, plus élevée que nécessaire, ne nuira, certes pas, au bon fonctionnement du système, mais engendrera une consommation de puissance inutilement élevée. De même dans le cas inverse où la température d'entrée est très élevée, le ventilateur à vitesse constante ne saurait s'adapter à cette hausse de la température. Il s'ensuivra alors que les serveurs ne seront pas suffisamment refroidis, entraînant dans l'immédiat, des points chauds dans le data center, et à long terme, une détérioration plus rapide des machines.

Afin de remédier à ce problème, deux autres types de ventilateurs à vitesse variable ont été également conçus. Le premier type permet une variation de la vitesse par morceaux et la puissance consommée augmente par plage de température, comme l'illustre la Figure 1.9. Le troisième type de ventilateur, présenté à la Figure 1.10, permet une variation continue

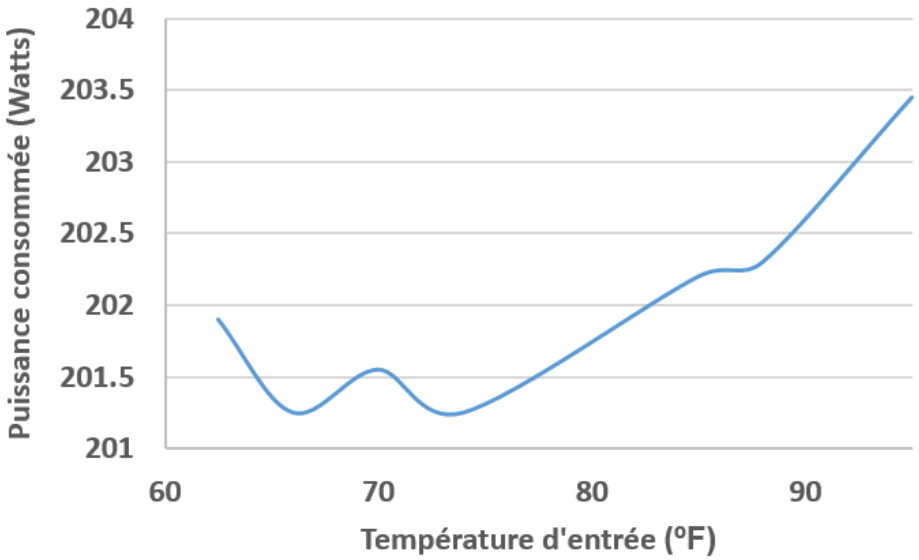


Figure 1.8 Puissance consommée en fonction de la température (Ventilateur de type “X”)

de la vitesse et de la puissance consommée sur l’intervalle de température considérée. Selon les lois régissant le principe de fonctionnement d’un tel type de ventilateur (Lee, 2012), la vitesse de rotation du ventilateur influence ses performances, notamment en termes de flux d’air pénétrant les serveurs et de puissance consommée par le ventilateur, lesquels varient respectivement au carré et au cube de la vitesse de rotation. Or, comme cette dernière augmente linéairement par rapport à la température d’entrée, la puissance consommée par

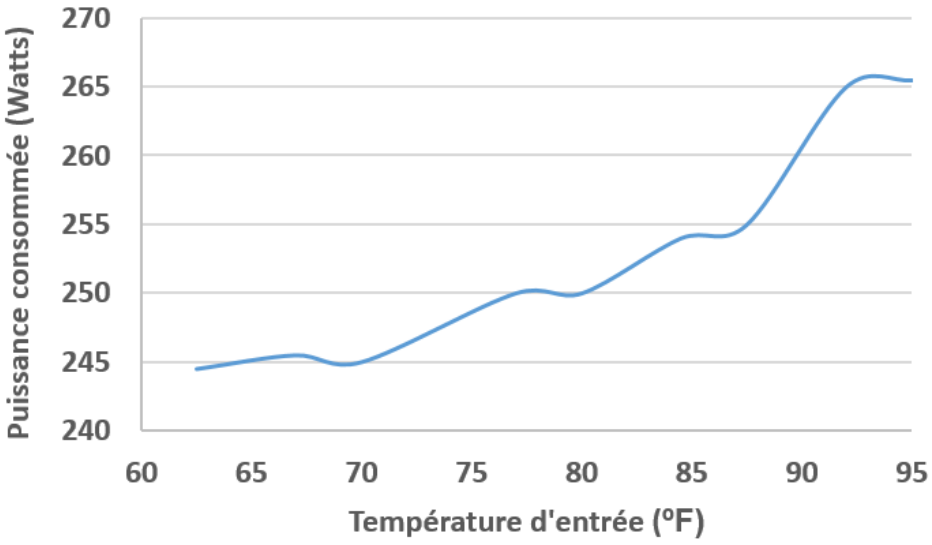


Figure 1.9 Puissance consommée en fonction de la température (Ventilateur de type “S”)

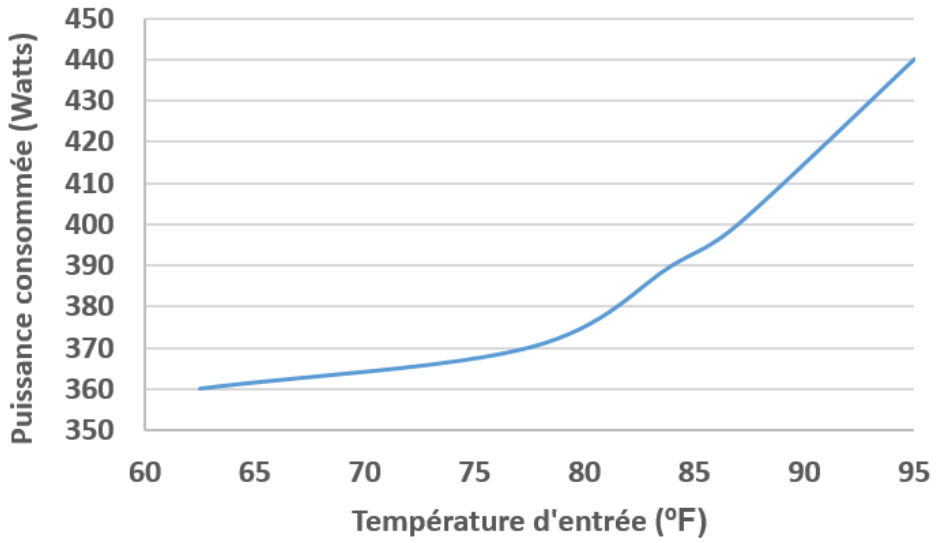


Figure 1.10 Puissance consommée en fonction de la température (Ventilateur de type “L”)

un ventilateur varie, elle aussi, au cube de la température, à une constante près, ce qui en fait la composante la plus influencée par la variation de la température d'entrée.

Serveurs Les serveurs constituent l'élément principal dans le contexte du Cloud Computing, facilitant l'hébergement des applications en fournissant la puissance de calcul nécessaire. L'énergie consommée par un serveur varie linéairement en fonction de la charge qui y est hébergée - quantité de CPU et de disque, mais n'en dépend pas uniquement. Comme l'illustre la Figure 1.11, environ 60% de la puissance totale d'un serveur est consommée en mode veille (en l'absence de charge), et est imputable à l'alimentation du serveur (Chen *et al.*, 2008).

Châssis Dans le modèle des serveurs-lame, un châssis est en général constitué d'un ensemble de serveurs alimentés par une même source et refroidis par les mêmes ventilateurs. De ce fait, la puissance consommée par un châssis comprend l'énergie en mode veille, associée à l'alimentation du châssis en tant que tel, la dissipation des ventilateurs qui y sont rattachés et la puissance consommée par les serveurs actifs (voir Pakbaznia et Pedram, 2009).

Commutateurs La consommation énergétique d'un commutateur suit un modèle identique à celui du serveur, où une certaine quantité de puissance est consommée en mode veille, et une autre partie est proportionnelle au trafic traversant le commutateur (voir Mahadevan *et al.*, 2009).

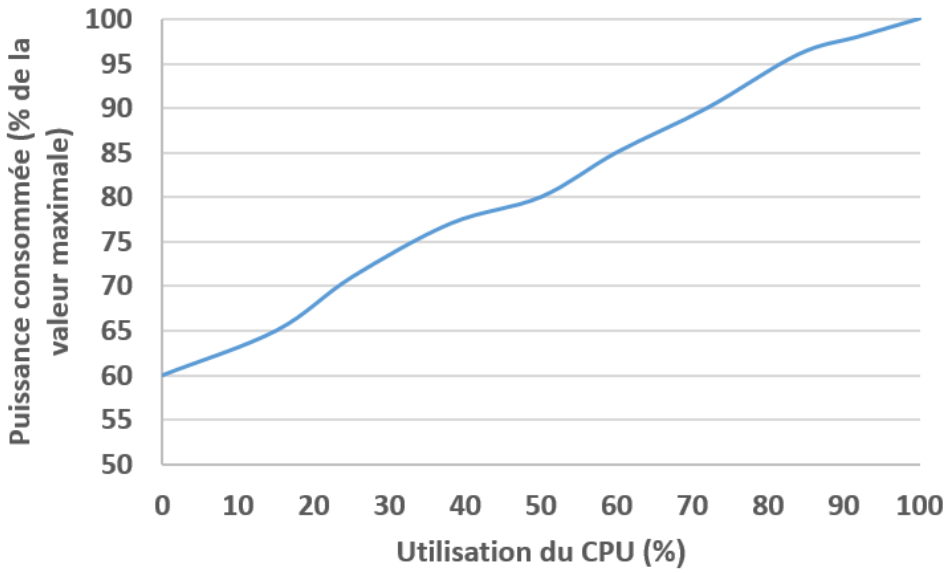


Figure 1.11 Variation de la puissance d'un serveur

1.1.4.4 Système de refroidissement ou CRAC

Un système de refroidissement se charge de maintenir une température adéquate à l'intérieur du data center, particulièrement lorsque les conditions ne permettent pas de profiter de la température extérieure (*free cooling*). De manière générale, le cycle de refroidissement, présenté à la Figure 1.12, se déroule comme suit : l'air chaud s'échappant des serveurs et quittant la pièce par des orifices situés au plafond, passe dans un échangeur thermique traversé par un réfrigérant - généralement de l'eau froide ; une fois refroidi, l'air frais est prêt à être acheminé dans la salle des machines, à travers les grilles installées au sol ; quant à l'eau chaude, elle retourne dans le refroidisseur pour être à nouveau rafraîchi (Pakbaznia et Pedram, 2009).

L'efficacité du système de refroidissement, définie par son coefficient de performance ou *Coefficient of Performance (COP)*, dépend de différents facteurs, comme le liquide utilisé dans le refroidisseur et la vitesse des pompes. Il se mesure comme étant le rapport entre la quantité de chaleur évacuée et la puissance consommée par le refroidisseur pour éliminer cette chaleur. En supposant que la puissance consommée par les équipements informatiques se transforme totalement en chaleur (Pelley *et al.*, 2009), le COP d'un système de refroidissement se quantifie en termes de ratio de la consommation énergétique des équipements, sur la puissance qu'il dissipe. Par ailleurs, ce coefficient de performance n'est pas constant et croît avec l'augmentation de la température de l'air frais fourni par le CRAC d'un data center (Pakbaznia et Pedram, 2009), comme l'illustre la Figure 1.13.

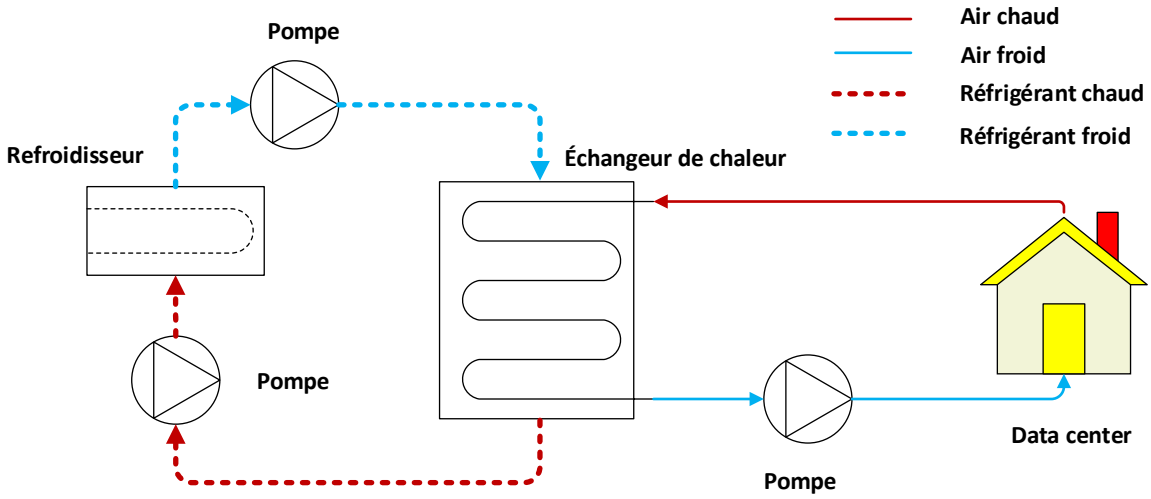


Figure 1.12 Système de refroidissement d'un data center (Posladek, 2008)

1.1.4.5 Performance énergétique

Afin de mieux évaluer les performances énergétiques des data centers, plusieurs métriques peuvent être considérées, notamment l'indicateur d'efficacité énergétique ou *Power Usage Effectiveness (PUE)* et le facteur d'émission ou *Greenness factor (GF)*.

Indicateur d'efficacité énergétique ou PUE Dans la lancée du mouvement en faveur de l'informatique éco-responsable ou *Green-IT*, le consortium *The Green Grid* (Avelar *et al.*,

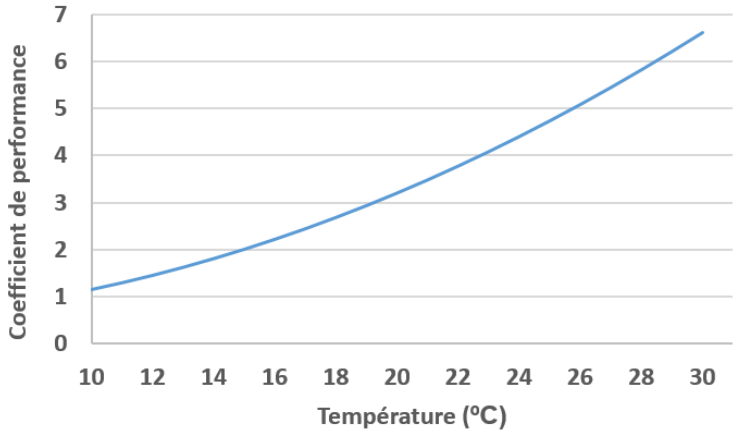


Figure 1.13 Évolution du COP en fonction de la température (Moore *et al.*, 2005)

2012), a introduit, en 2007, un nouveau facteur permettant d'évaluer les performances d'un data center, le PUE. Depuis, ce facteur s'est érigé en tant qu'outil officiel de référence international, en termes de mesure d'efficacité énergétique, permettant aux fournisseurs de Cloud de concevoir des data centers de moins en moins énergivores. Le PUE se définit comme le rapport entre l'énergie totale consommée par un data center et celle dépensée uniquement par le fonctionnement des équipements informatiques. De manière générale, le PUE idéal, égal à 1.0, signifierait que la consommation énergétique d'un data center est totalement imputable à l'équipement informatique déployé, toutefois, en raison de la présence d'équipements électriques, ce ratio est toujours supérieur à l'unité. Par ailleurs, comme l'indique la Figure 1.7, le système de refroidissement étant l'équipement électrique le plus énergivore, toute tentative d'augmentation de l'efficacité énergétique doit forcément découler d'une réduction de la consommation de ce dernier.

Facteur d'émission ou Greenness factor Dans le domaine de la pollution atmosphérique, l'impact environnemental d'une composante quelconque est évalué en considérant son facteur d'émission. Ce dernier se définit comme le ratio entre la quantité de GES rejetés dans l'air par cette entité, et la mesure de la métrique qui la caractérise, comme la quantité de CO_2, en grammes par kilomètre parcouru, dans l'industrie du transport. Ce facteur permet d'évaluer le bilan carbone - ou quantité de polluants atmosphériques - résultant d'une activité quelconque. Dans le monde des télécommunications, il permet de caractériser l'importance des émissions liées à la production de l'énergie utile et s'exprime en grammes de CO_2 rejetés par unité d'énergie ou de puissance consommée (Moghaddam *et al.*, 2011). Or, dans l'univers du Cloud Computing, les data centers étant en général alimentés par plusieurs sources d'énergie, renouvelables ou non, une source à la fois, le facteur d'émission dudit data center devient alors dépendant, non seulement du facteur d'émission des sources d'énergie utilisées, mais aussi de leur fréquence d'utilisation. Dans ce cas de figure, le facteur d'émission dudit data center est une valeur moyenne calculée comme la somme du facteur d'émission de chaque source d'énergie alimentant le data center, pondéré par le pourcentage d'utilisation de chacune de ces sources.

1.1.5 Processus global d'hébergement des applications dans le Cloud

Le processus d'hébergement des applications dans le Cloud, illustré à la Figure 1.14, se divise en trois (3) grandes phases, et fait intervenir trois acteurs principaux : des clients, un négociateur et des fournisseurs d'infrastructure.

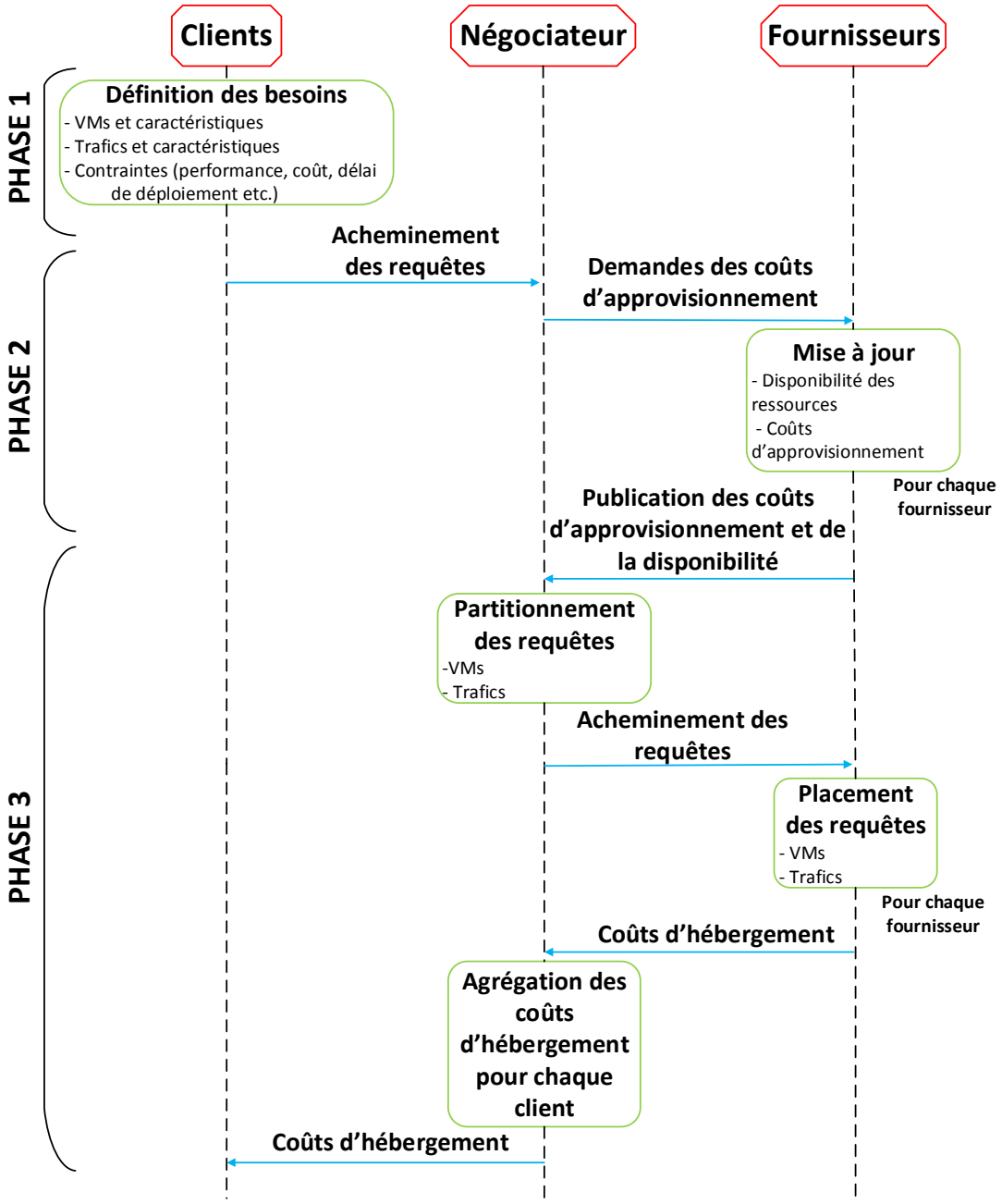


Figure 1.14 Processus d'hébergement des applications dans le Cloud

1.1.5.1 Phase 1

Dans la première phase, les clients traduisent leurs applications en demandes en termes de VMs et de trafic associé, en définissant particulièrement les besoins en puissance de calcul, de mémoire, de stockage et de bande passante. Ils déterminent également certaines contraintes

de performance ou de localisation de leurs applications. Ces spécifications, VMs, trafics et contraintes, sont ensuite acheminées à un négociateur.

1.1.5.2 Phase 2

À la seconde phase, le négociateur se charge de sélectionner les fournisseurs de Cloud les plus adéquats, permettant de satisfaire les besoins des clients. Ce choix des fournisseurs se fait, en général, en considérant la disponibilité des ressources et les coûts d'approvisionnement publiés par chaque fournisseur. Une fois le processus de sélection terminée, les demandes sont réparties entre les fournisseurs choisis.

1.1.5.3 Phase 3

Dans la dernière phase, chaque fournisseur s'adonne au processus de placement des applications à l'échelle de leurs data centers. Le coût final d'hébergement est, par la suite, retourné par chaque fournisseur au négociateur qui, à son tour, agrège les coûts et renvoie le montant total aux clients.

1.1.6 Placement des applications dans le Cloud

La troisième phase, l'assignation des applications à l'infrastructure physique, est l'étape la plus complexe et constitue le sujet de cette thèse. Dans cette phase, le processus de placement s'effectue de manière itérative et comprend plusieurs étapes, comme présenté à la Figure 1.15.

Une première phase consiste à déterminer la demande à héberger. À cette étape, les informations relatives aux applications à héberger, nouvelles ou en attente, sont collectées. Ces données font référence au nombre et aux caractéristiques des VMs à placer et du trafic à router dans le réseau. Cette étape permet également d'évaluer les exigences des clients, lesquelles sont par la suite traduites en contraintes de placement.

Suit alors la phase de collecte des informations en rapport avec l'état de l'infrastructure physique et des applications déjà hébergées. Des détails concernant les caractéristiques et l'utilisation des équipements, la quantité de ressources disponibles, la durée de vie restante des applications hébergées ou encore, la variation en termes de besoins des applications, sont autant d'informations agrégées à cette étape. Dans le contexte d'un fournisseur Cloud devant gérer plusieurs data centers, ces informations sont transmises par une entité, le coordonnateur de Cloud ou *Cloud coordinator*. Ce dernier se charge d'assurer la gestion et la coordination des différentes activités internes au Cloud, comme l'ordonnancement des tâches, les mécanismes d'allocation des VMs, la gestion de la virtualisation, le contrôle et la surveillance des

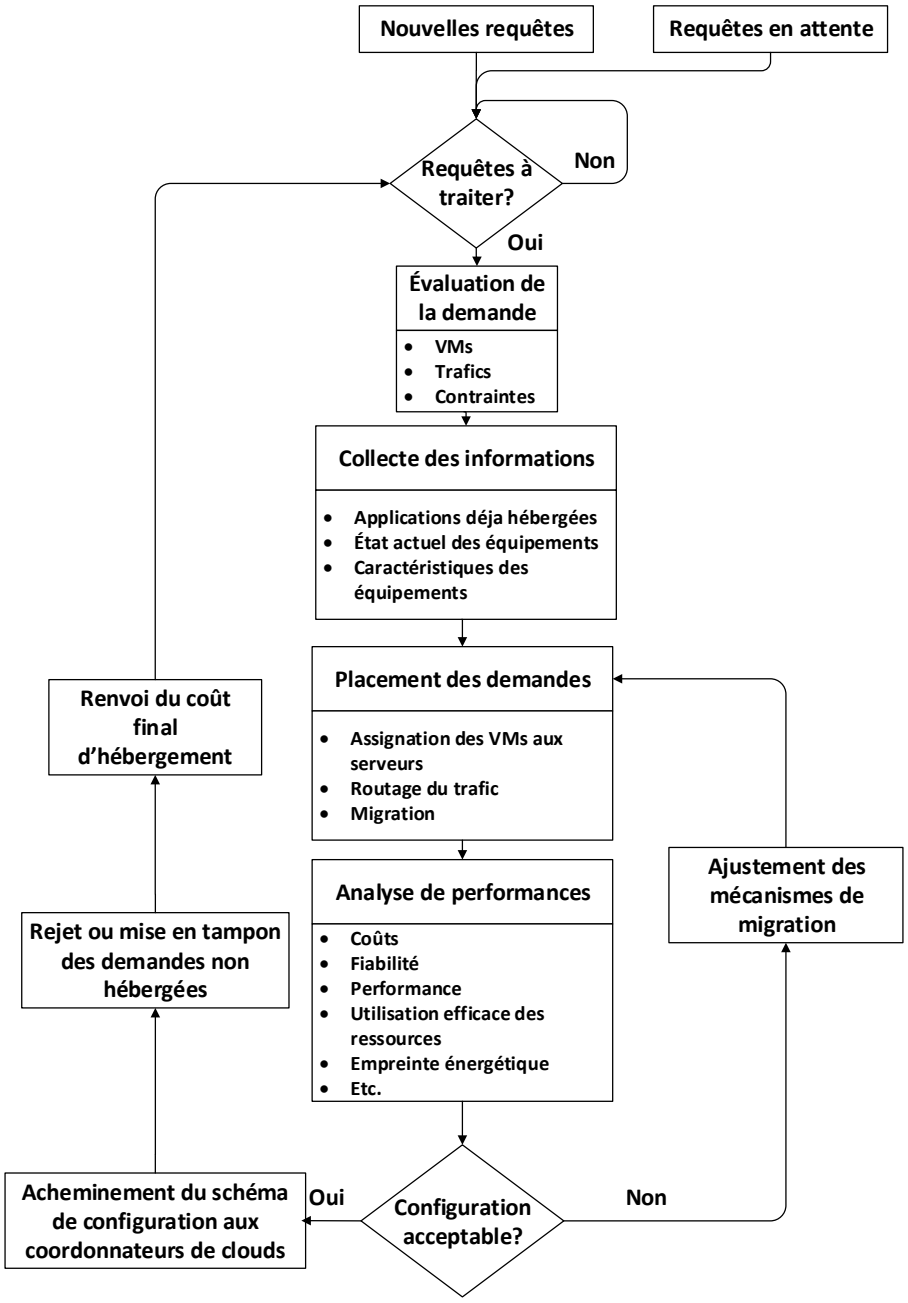


Figure 1.15 Processus d'assignation des applications à l'infrastructure physique

applications hébergées (*monitoring*).

L'ensemble de ces données définissant les demandes et l'état actuel de l'infrastructure physique servent d'entrée à la phase d'assignation des VMs et du trafic. Plus particulièrement, dans cette phase, les fournisseurs tentent d'assigner les VMs aux serveurs adéquats et à router

le trafic dans le réseau, tout en considérant les limites physiques des équipements et les requis, en termes de performance et de localisation des applications, exigés par les clients. Certaines politiques de migration des applications peuvent être également mises en place afin d'assurer un placement efficace. De manière générale, un critère d'optimisation particulier, tels l'utilisation efficace des ressources physiques, le coût d'exploitation, l'empreinte énergétique etc., permet de guider le processus de placement des demandes.

À l'étape d'évaluation, les objectifs et les critères de performance sont évalués et, dans le cas d'une configuration non satisfaisante, les mécanismes de migration sont ajustés avant de retourner à la phase de placement. Suite à l'obtention d'un schéma de placement acceptable, les informations relatives à la nouvelle configuration sont acheminées à chaque *Cloud coordinator* qui se charge de déployer les applications dans chaque data center. Aussi, parallèlement, les demandes - VMs et trafics - non hébergées sont, soit mises en tampon, en attente de disponibilité, ou encore complètement rejetées. Enfin, le coût réel d'hébergement est renvoyé au négociateur et le processus retourne à la case départ de collecte d'informations.

Une fois les concepts de base se rapportant au Cloud Computing aient été définis, vient le moment de faire état des principaux problèmes rencontrés lorsqu'un fournisseur, propriétaire d'un InterCloud, veut placer des applications, VMs et trafics, à l'échelle de plusieurs data centers géographiquement distribués, dans le but de réduire son empreinte environnementale.

1.2 Éléments de la problématique

Dans le contexte du modèle de service IaaS, bien que les offres d'hébergement des applications soient de plus en plus nombreuses, le processus d'allocation de ces dernières aux data centers demeure un problème relativement complexe à solutionner.

Ce qui fait la force de ce nouveau paradigme informatique, soit la disponibilité des ressources physiques, en constitue également une grande faiblesse, particulièrement au niveau du processus d'allocation des VMs et du trafic associé. Contrairement à la mise en place d'un réseau physique où le concepteur est libre de faire certains choix, dans le cas de l'assignation des applications, l'enjeu principal, pour le fournisseur, revient à devoir conjuguer avec l'infrastructure physique sous-jacente, déjà existante, en développant des mécanismes afin de maximiser l'utilisation de l'éventail des moyens qui s'offrent à lui. Ce manque certain de latitude ajoute un degré de difficulté au processus d'assignation des applications, car le fournisseur est contraint de se limiter à la quantité de ressources disponibles.

Dans le contexte particulier de l'optimisation de l'empreinte carbone à l'échelle d'un InterCloud, où les data centers sont alimentés par différentes sources d'énergie, renouvelables ou

non, la minimisation de l'énergie ne saurait nécessairement traduire une réduction de l'empreinte carbone. En effet, dans le cas d'un data center alimenté par une énergie verte, la consommation énergétique peut être très élevée alors que l'empreinte carbone est quasiment nulle, tandis que lorsque la source d'énergie est semi ou non verte, l'impact écologique devient intimement lié à la consommation énergétique. Dans ce dernier cas de figure, minimiser l'énergie consommée induirait une réduction de l'empreinte carbone dudit data center.

Par ailleurs, ce problème de consommation énergétique excessive s'explique, en général, par le fait que, dans le but de faire face aux besoins sans cesse fluctuants des utilisateurs, les sites d'hébergement se tiennent toujours prêts en gardant presque la totalité des ressources en état d'attente latente, alors que, la plupart du temps, seulement 50% des capacités de calcul sont réellement sollicitées. Ceci conduit à un gaspillage qui peut facilement s'élever à environ 90% de l'énergie consommée. Aussi, bien qu'à la puissance des serveurs s'ajoute également la consommation non négligeable des ressources réseau et du système de refroidissement, dans le but de réduire l'impact énergétique des data centers non verts, l'approche la plus courante consiste à s'attaquer à un sous-problème à la fois, afin de contourner la complexité du problème global.

Dans ce contexte, les premiers efforts ont porté sur les méthodes de consolidation des VMs afin de minimiser le nombre de serveurs actifs. Le processus s'assimile alors à un problème de Bin Packing (Garey et Johnson, 1990). Toutefois, compte tenu de l'hétérogénéité des équipements dans un data center, cette approche ne se révèle pas être viable, et le problème de réduction de l'énergie des nœuds de calcul a été attaqué en considérant les caractéristiques énergétiques de ces derniers, l'objectif étant de réduire, de préférence, la puissance totale consommée par ces équipements.

Par ailleurs, alors que le concept de virtualisation des serveurs semble permettre une économie d'énergie, en facilitant la consolidation de plusieurs VMs sur un même support physique, il est également à l'origine de différents problèmes lorsque vient le temps de procéder à ce regroupement des VMs. En effet, sachant qu'une consolidation aveugle peut engendrer des conflits lors de l'accès aux ressources physiques, entraînant ainsi une dégradation de performance des applications consolidées, une augmentation de leur durée d'exécution et de l'énergie consommée, la principale difficulté de ce processus revient à déterminer l'emplacement idéal pour chaque type de machine en question. À cet effet, il devient primordial d'identifier les particularités de l'application s'exécutant sur la VM, de manière à déterminer le type et l'importance des ressources nécessaires au bon fonctionnement de cette dernière, de même que les paramètres de performance auxquels elle est particulièrement sensible. De plus, avec le Cloud Computing, et particulièrement l'externalisation des ressources, des problèmes

relatifs à la sécurité, à la territorialité et à la fiabilité peuvent également survenir. En ce sens, les clients, soucieux quant à l'intégrité et à la confidentialité de leurs informations, peuvent également imposer des contraintes de co-hébergement ou de localisation de leurs applications.

Avec le cooling qui représente environ 30% de la consommation énergétique, nombreux sont les travaux qui ont tenté d'optimiser l'efficacité énergétique à l'intérieur d'un data center. Plusieurs techniques de gestion thermique ont donc été mises en place afin de réduire l'impact du cooling sur la facture énergétique dans un data center, allant de la réduction du taux de recirculation de la chaleur, à la minimisation des points chauds. Plus récemment, l'American Society of Heating, Refrigerating and Air Conditioning Engineers (ASHRAE) a démontré qu'une manière simple et efficace de réduire la puissance du CRAC consistait à augmenter la température fournie par ce tel système, jusqu'aux limites de fonctionnement autorisées pour les équipements informatiques. Toutefois, cette technique ne prend pas en compte le comportement dynamique des ventilateurs des serveurs dont la puissance augmente au cube de la température fournie par le CRAC, ce qui peut, à un certain point, annuler les gains d'énergie associés au cooling à température élevée.

Par ailleurs, la grande popularité du Cloud facilite l'hébergement d'un grand éventail d'applications, allant des applications encapsulées dans une unique machine, aux applications plus complexes, se répartissant sur plusieurs VMs distinctes. Dans le contexte de ces dernières applications, l'interaction entre les entités composant ces applications introduit un trafic de données non négligeable dans le réseau, entraînant ainsi une consommation énergétique des équipements réseau, et particulièrement des commutateurs. Ainsi, à l'instar des travaux antérieurs portant sur la puissance des serveurs, plusieurs chercheurs ont tenté de résoudre le problème de consommation énergétique dans le réseau en essayant d'abord de réduire le nombre de commutateurs utilisés. Dans un second temps, en raison des différents types d'équipements, ils ont de préférence abordé la question en considérant le profil énergétique de ces éléments.

En général, ces problèmes sont traités de manière isolée avec l'objectif d'intégrer les solutions partielles au problème global. Toutefois, la méthode d'optimisation séquentielle et/ou isolée ne saurait s'appliquer dans le contexte du placement des applications. En effet, l'approche séquentielle ne peut être utilisée, car aucune hiérarchie, quant à l'ordre d'optimisation des différentes phases, impliquant que la configuration issue de l'optimisation d'une étape antérieure serve d'entrée à la phase suivante, n'a été définie, à l'instar des réseaux classiques. Quant à l'optimisation isolée, elle suppose l'existence de mécanismes d'intégration des résultats issus des différentes phases. Toutefois, non seulement de tels mécanismes n'existent pas actuellement, mais le processus d'intégration peut se révéler impossible du fait que l'optimisation

isolée peut résulter en des configurations incompatibles.

D'autre part, comme démontré dans les réseaux classiques (St-Hilaire *et al.*, 2006), il existe diverses interactions qui ne sont pas considérées lorsque les problèmes sont résolus individuellement. Plus particulièrement, dans le cas de la minimisation de l'énergie totale, une optimisation de la puissance des nœuds de calcul peut conduire à une configuration introduisant un routage de trafic sous-optimal ; de même, l'optimisation de l'impact des ressources réseau peut résulter en des schémas non faisables, où le placement des VMs devient impossible en raison des limites des ressources physiques. Quant à l'optimisation de l'efficacité énergétique dans un data center, une augmentation de la température interne ne saurait se faire sans tenir compte du comportement dynamique des ventilateurs face à la variation de température, car un tel mécanisme aveugle peut aller jusqu'à annuler les gains liés à l'économie du cooling à haute température. De plus, face au problème d'empreinte carbone à l'échelle d'un InterCloud, le problème devient encore plus complexe, car en plus de la puissance totale de chaque data center, il importe également de considérer le facteur d'émission de chacun. Considérant la nature hétérogène des équipements, le processus d'assignation ne consiste pas à placer les VMs et le trafic associé dans les data centers les plus verts d'abord, mais de considérer le problème dans son ensemble en assignant les applications aux équipements les plus éco-énergétiques, via une consolidation intelligente, tout en respectant les exigences de performance et de sécurité imposées par les clients.

Le problème de placement des VMs et de leur trafic dans un InterCloud, dans le but de minimiser l'empreinte écologique d'un tel environnement est un problème difficile à résoudre. Bien que la décomposition en sous-problèmes semble être moins complexe à mettre en œuvre, aucun mécanisme d'intégration n'a jamais été développé, sans oublier que ce dernier ne nous assurerait pas forcément de l'optimalité de la solution finale. Ainsi, la planification des applications est, en réalité, beaucoup plus complexe qu'elle ne semble le paraître, et seule une optimisation conjointe des différents sous-problèmes permettrait d'obtenir la configuration de coût minimal.

Par ailleurs, ce mécanisme de placement des applications se doit d'être un processus continu, en raison de la variabilité des charges. Naît alors la nécessité de développer des mécanismes automatiques d'assignation des applications aux data centers. De manière générale, différentes méthodes de résolution sont utilisées pour solutionner ce genre de problème. Des approches automatiques ont été privilégiées au détriment de la résolution manuelle, car, en raison des mises à jour et surtout des problèmes de grande taille, il est très difficile, voire même impossible, d'évaluer l'éventail de toutes les possibilités à la main. Pour ce qui est des méthodes exactes, elles ont la particularité d'aboutir à la solution optimale, cependant, le temps de

calcul peut être extrêmement long. Une manière efficace de répondre à ces problèmes revient donc à développer des heuristiques qui permettent d'avoir un bon compromis entre la qualité des solutions et le temps d'exécution.

De ce fait, afin de tenir compte des différents enjeux auxquels il est important de faire face dans le processus d'assignation des applications aux data centers, une optimisation conjointe est fortement suggérée. Ce modèle global tendrait à ajouter un peu plus de réalisme au problème de localisation des applications dans les data centers, car un placement optimal ne peut s'effectuer efficacement sans tenir compte des éléments influençant l'impact environnemental des data centers, tout en considérant les interactions, conflictuelles ou non, pouvant exister entre ces entités, ce qui amène obligatoirement à effectuer certains compromis. Plus concrètement, dans le but de minimiser l'empreinte carbone dans un environnement InterCloud, les questions relatives aux data centers à considérer, à la valeur optimale de leur température de fonctionnement, aux VMs à consolider, aux performances et contraintes de localisation à intégrer, ou encore au trafic à router, doivent être analysées. Ainsi, la mise en œuvre d'un tel modèle global est un travail assez laborieux, mais aura le mérite de fournir d'excellentes solutions.

Les divers éléments de la problématique énoncés ci-dessus nous ont donc amenés à nous poser les questions suivantes :

- comment modéliser, à l'aide d'équations mathématiques, la consommation énergétique à l'intérieur d'un data center nous permettant de considérer les interactions conflictuelles entre les différents éléments énergivores ?
- notre modélisation est-elle en mesure de bien refléter la réalité du processus de placement des applications dans le Cloud, en considérant les différentes exigences liées aux applications, afin d'éviter toute dégradation de performance ?
- comment modéliser les contraintes de co-hébergement et de localisation relatives à la sécurité, à la territorialité et à la redondance des applications ?
- sommes-nous en mesure de proposer un modèle de programmation mathématique global permettant de placer des VMs et leur trafic dans un environnement InterCloud, en minimisant l'empreinte écologique d'un tel environnement, tout en tenant compte des exigences des clients ?
- saurons-nous mettre en œuvre des méthodes de résolution approximatives nous permettant d'obtenir un excellent compromis entre le temps d'exécution et la qualité des solutions obtenues ?

Ainsi, nombreuses sont les questions qui nous ont portés à énoncer les principaux objectifs de recherche présentés à la section suivante.

1.3 Objectifs de recherche

Cette thèse vise, principalement, à concevoir un cadre de planification globale des machines virtuelles et de leur trafic dans la perspective d'une minimisation de l'empreinte carbone dans un environnement InterCloud. Dans le contexte d'un processus de placement statique des applications, l'optimisation conjointe de plusieurs composantes permettra au fournisseur Cloud de considérer toutes les interactions existant entre ces entités, afin de déterminer la configuration de coût minimal, tout en tenant compte des exigences des clients. De manière plus spécifique, cette thèse vise à :

- Concevoir un modèle mathématique pour l'assignation des VMs aux serveurs, dans le but de réduire l'impact écologique d'un InterCloud. Plus particulièrement, il s'agira de combiner le facteur d'émission des data centers à un modèle énergétique qui inclut la consommation des châssis et du système de refroidissement ainsi que le comportement dynamique des équipements. Il sera également question d'y intégrer les notions relatives à la dégradation de performance des applications, et la modélisation des contraintes de co-hébergement des VMs, afin de considérer les requis, en termes de sécurité et de redondance, imposés par le client.
- Évaluer l'importance relative à la détermination de la température optimale de chaque data center actif, par le biais d'une étude analytique de la fonction objectif.
- Concevoir un modèle de programmation mathématique permettant de résoudre, de manière exacte, le problème de placement des VMs dans un environnement InterCloud afin de le comparer aux approches de référence citées dans la littérature.
- Proposer une méthode de résolution basée sur les métaheuristiques afin de résoudre les problèmes de grande taille avec le modèle d'empreinte carbone proposé.
- Évaluer les performances de l'heuristique proposée, en termes de bon compromis entre vitesse d'exécution et qualité des solutions obtenues, par rapport à une borne inférieure et par rapport à d'autres méthodes de résolution approximatives.
- Étendre le modèle mathématique afin d'intégrer l'assignation des applications complexes, plus particulièrement le routage du trafic inter-VMs et les concepts liés à la localisation de ces applications, en termes de data center potentiels.
- Proposer et évaluer une méthode de résolution hybride basée sur les métaheuristiques afin de résoudre les instances de grande taille avec le modèle global d'empreinte carbone proposé.

1.4 Principales contributions de la thèse et leur originalité

La grande originalité de cette thèse repose sur la conception d'un cadre de résolution du problème de placement des charges, afin de minimiser l'impact écologique d'un InterCloud. En ce sens, les principales contributions portent autant sur la modélisation du processus de placement des charges, que sur le développement de méthodes visant à solutionner un tel problème, et peuvent être détaillées comme suit :

1. **Développement d'un modèle plus précis de consommation énergétique et d'empreinte carbone.** Une première innovation consiste à proposer une nouvelle évaluation de l'énergie consommée par les data centers, en considérant des charges simples, à VM unique. Plus particulièrement, cette approche améliore les modèles existants en combinant la dissipation des nœuds de calcul, serveurs et châssis, et du système de refroidissement (CRAC), tout en considérant l'hétérogénéité des équipements. L'originalité de cette contribution repose également sur l'intégration de la nature dynamique des équipements de calcul face à la température. Cette considération est d'autant plus importante, car elle permet de mieux cerner le comportement antagoniste des équipements informatiques et du CRAC, face à une augmentation de la température ambiante. Ce modèle de consommation a été combiné au facteur d'émission des data centers afin de déduire un modèle d'empreinte carbone pour un environnement InterCloud. Nous avons également pu, par le biais d'une étude analytique, mettre en évidence l'importance associée à l'évaluation de la température optimale de chaque data center actif, soulignant ainsi la pertinence du modèle de consommation proposé.
2. **Nouvelle modélisation des exigences des clients.** Afin de mieux prendre en compte les requis des clients, aux exigences liées aux performances des applications, ont été ajoutées des contraintes visant à assurer la confidentialité et une certaine redondance de la charge hébergée. Cet ajout est d'autant plus pertinent, car en dépit de l'externalisation des données, les clients détiennent toujours un certain droit de regard quant à la sécurité de leurs applications. D'où la pertinence de développer un modèle capable d'intégrer les exigences des clients, peu importe leur nature.
3. **Modèle de programmation mathématique visant à minimiser l'empreinte carbone d'un InterCloud.** Nous avons proposé un modèle, basé sur les techniques de programmation en nombres entiers, et permettant simultanément de déterminer l'emplacement des VMs et la température à fournir à l'intérieur de chaque data center actif, de manière à équilibrer la puissance du CRAC et la consommation des châssis et serveurs. Il s'agit d'une avenue de recherche assez révolutionnaire, car, dans l'optique de réduire l'empreinte écologique d'un InterCloud, au lieu de regarder le problème

de placement des VMs et celui de l'augmentation de l'efficacité énergétique avec des œillères, cette approche, beaucoup plus complète, permet de considérer le problème dans sa globalité, intégrant ainsi les complexes interactions entre les principaux facteurs de consommation énergétique et environnementale.

4. **Adaptation d'une méthode heuristique pour résoudre les grandes instances du problème.** Un autre côté original à notre thèse se situe au niveau de la méthode de résolution. En effet, lorsque la taille du problème augmente considérablement, les mécanismes basés sur la résolution exacte deviennent moins intéressants, car ils entraînent souvent des temps d'exécution exponentiellement élevés et des problèmes de dépassement de la mémoire disponible. De là nous est venue l'idée de développer une heuristique basée sur la Recherche Locale Itérée ou *Iterated Local Search (ILS)*. Cette méthode est jugée utile car, particulièrement pour les instances de grande taille, elle permettra au planificateur d'obtenir de bonnes solutions en un temps raisonnable.
5. **Extension du modèle mathématique afin de considérer différents types de charges à placer.** Outre les charges à VM unique, on retrouve également des applications plus complexes, nécessitant l'interaction entre plusieurs VMs, et induisant un trafic de données non négligeable en termes de consommation énergétique dans le réseau. Nous avons donc enrichi notre modèle mathématique de manière à intégrer la prise en charge d'applications plus complexes, dans le processus de placement. Plus spécifiquement, ce nouveau modèle permet d'optimiser conjointement le placement des VMs, l'efficacité énergétique et le routage du trafic en sélectionnant les équipements (châssis, serveurs et commutateurs) les plus éco-énergétiques, dans la perspective d'une optimisation de l'empreinte carbone d'un environnement InterCloud.
6. **Développement d'une méthode de résolution basée sur l'hybridation d'approches heuristiques.** Notre dernier apport, un algorithme général de placement des applications, est d'autant plus intéressant, car il représente une contribution autant dans le monde de la planification que dans le domaine des métaheuristiques. En effet, il considère le problème de réduction de l'empreinte carbone comme un tout, en plaçant les VMs, en acheminant le trafic inter-VMs et en optimisant l'efficacité énergétique, toujours sous la contrainte des exigences des clients. Outre cette contribution, la grande originalité de cette méthode réside dans l'adaptation d'une approche peu connue, l'hybridation de plusieurs métaheuristiques, où les particularités des différentes méthodes ont été combinées et largement exploitées dans le but de mieux explorer l'espace de recherche et obtenir de bonnes solutions en un temps polynomial.

De manière générale, en utilisant ces différents outils, les fournisseurs Clouds se retrouveront mieux assistés dans leur lourde tâche de déterminer l'emplacement optimal de chaque VM et le routage idéal du trafic associé, dans le but ultime de réduire l'important fardeau écologique imputé à leur infrastructure informatique.

1.5 Plan de la thèse

Après avoir, dans ce chapitre, défini les concepts de base, énoncé la problématique et les objectifs de recherche, et souligné les principales contributions de cette thèse, le chapitre 2 présentera les principaux travaux relatifs à l'optimisation de l'empreinte carbone dans un environnement InterCloud via le processus de placement des applications. Plus particulièrement, il s'agira de faire état de la littérature en ce qui a trait à la minimisation de l'énergie dans un data center, aux efforts déployés pour réduire l'impact écologique d'un ensemble de Clouds, et aux approches utilisées pour résoudre les problèmes difficiles. Une analyse des travaux sera ensuite effectuée en vue de mettre en évidence les failles de ces approches par rapport à la problématique soulevée. Quant au chapitre 3, il décrira les démarches de l'ensemble du travail de recherche, en démontrant l'étroite relation entre les objectifs énoncés à la Section 1.3 et les articles réalisés dans le cadre de cette thèse.

Le chapitre 4 présente le texte intégral d'un article intitulé « *On the Carbon Footprint Optimization in an InterCloud Environment* ». Cet article propose un modèle de placement des VMs dans un InterCloud, dans un contexte d'optimisation de l'empreinte carbone dudit environnement. Plus particulièrement, ce modèle combine le facteur d'émission à la puissance dépensée, où la consommation des nœuds de calcul et la dissipation du cooling sont simultanément évaluées, tout en considérant les notions de performances et de sécurité. Une particularité de ce modèle repose sur l'intégration du comportement dynamique des ventilateurs influençant la température de compromis entre la puissance des châssis et les gains d'énergie associés au cooling. L'importance de la détermination de la température optimale pour chaque data center actif a été démontrée à travers une analyse de la monotonie et de la convexité du modèle. Un algorithme basé sur la méthode de résolution exacte a été proposé pour ce modèle, dont la pertinence, ainsi que celle de l'intégration des contraintes de consolidation, ont été mises en évidence dans ce travail. Cet article a été accepté pour publication dans la revue *IEEE Transactions on Cloud Computing*.

Le chapitre 5 présente le texte intégral d'un article intitulé « *An Iterated Local Search Approach for Carbon Footprint Optimization in an InterCloud Environment* ». Dans ce dernier, une méthode approchée, basée sur les métaheuristiques, a été développée pour résoudre le problème d'optimisation de l'empreinte carbone dans un environnement InterCloud. Cette

méthode a été mise en place pour solutionner les grandes instances du problème ne pouvant être résolues à l'aide de la méthode exacte. Les résultats présentés dans cet article démontrent que l'algorithme proposé, comparé à la méthode exacte ou à d'autres méthodes de résolution approchées, permet d'obtenir de bonnes solutions en un temps raisonnablement réduit. Cet article a été accepté pour publication dans la revue *International Journal of Metaheuristics*.

Le chapitre 6 présente le texte intégral d'un article intitulé « *A Hybrid Approach for Optimizing Carbon Footprint in InterCloud* ». Celui-ci propose un modèle global d'optimisation de l'empreinte carbone, lequel constitue une extension du modèle développé dans le premier volet de notre travail. Plus précisément, dans le processus de minimisation de l'empreinte carbone, en plus de considérer les aspects mis en évidence au chapitre 4, ce modèle intègre également le placement des applications complexes et les contraintes de localisation associées, lesquelles applications nécessitent des interactions entre plusieurs VMs. Ainsi, ce modèle, plus complet, réalise une optimisation conjointe du placement des VMs et du routage du trafic, permettant de considérer simultanément la puissance consommée par les nœuds de calcul, par les ressources réseau et par le système de refroidissement. Ce modèle, découlant de la composition de plusieurs problèmes NP-complets, est encore plus difficile à résoudre. Une métaheuristique hybride a donc été également proposée dans cet article afin de résoudre le problème d'optimisation de l'empreinte carbone. Les avantages d'une optimisation conjointe, ainsi que les performances de la méthode de résolution, en termes de compromis entre la qualité des solutions et le temps d'exécution, ont été mis en évidence dans cet article. Ce dernier a été soumis à la revue *IEEE Transactions on Services Computing*.

Le chapitre 7 présente les résultats qui n'ont pas pu être intégrés aux articles, en raison de règles éditoriales, limitant le nombre de pages. Le chapitre 8 propose une analyse générale, en regard des choix technologiques et méthodologiques effectués dans le cadre de cette recherche. Finalement, le chapitre 9 vient clore cette thèse, d'une part, en présentant un récapitulatif des travaux, soulignant ainsi l'originalité du travail réalisé, et d'autre part, en discutant des limitations et des potentielles avenues de recherche qui s'y rapportent.

CHAPITRE 2 REVUE DE LITTÉRATURE

Dans cette section, il sera question de présenter les principaux travaux qui ont été réalisés afin de minimiser l'énergie et l'empreinte carbone d'un environnement InterCloud, particulièrement dans le domaine du processus d'hébergement des applications dans le Cloud. Ce mécanisme vise, de manière générale, à déterminer un schéma d'assignation optimale des charges à l'infrastructure sous-jacente, sous la contrainte de la disponibilité des ressources physiques. Toutefois, outre un mécanisme de consolidation idéale, certaines particularités liées au processus d'assignation des applications sont à considérer, particulièrement lorsque vient le temps d'optimiser la puissance consommée et l'impact écologique d'un ensemble de data centers géographiquement distribués. Ainsi, avant de passer en revue les contributions majeures dans le domaine, il importe d'abord de décrire globalement un tel processus. Par la suite, les principales réalisations seront exposées, suivies des méthodes de résolution couramment utilisées. Enfin, une analyse des travaux présentés viendra clore ce chapitre.

2.1 Analyse sommaire du problème

Alors qu'il a toujours été assimilé au problème de *Bin Packing* ou de *Sac à Dos*, le processus d'assignation des applications aux data centers est en réalité beaucoup plus complexe à résoudre. En effet, une configuration optimale doit non seulement se baser sur la disponibilité physique des équipements, mais doit également incorporer de nouveaux éléments, comme le coût des équipements ou leurs caractéristiques énergétiques, en fonction du critère d'optimisation considéré. Par exemple, afin de réduire la puissance consommée, il importe également de tenir compte du profil énergétique des différents équipements lors du processus de consolidation. De plus, la prise en compte des exigences du client, en termes de localisation de leurs applications, ou encore de la nature des charges à consolider afin d'éviter toute forme de dégradation de performance, ajoute un autre degré de difficulté au problème.

Un autre aspect qui rend le processus encore plus ardu est que, dans le cas des applications complexes, il importe, non seulement d'assigner des VMs à des serveurs, mais également de considérer l'aspect de routage du trafic dans le réseau. Il est souvent de coutume de placer d'abord les VMs, ensuite d'acheminer le trafic à travers le réseau. Toutefois, compte tenu du niveau d'inter-relations entre ces deux entités (Jiang *et al.*, 2012), ces mécanismes ne peuvent s'effectuer de manière séquentielle. Dans un tel cas, une optimisation conjointe est de mise afin d'obtenir une configuration optimale.

Dans le cas particulier de la minimisation de l'énergie dans un data center, une troisième dimension, en rapport avec la consommation du système de refroidissement, se doit également d'être considérée. Une fois de plus, cet aspect ne saurait être optimisé indépendamment des autres, VMs et trafic, au risque d'ignorer les interactions entre ces différents éléments. Pour ce qui est de l'impact environnemental d'un ensemble de data centers géographiquement distribués, le facteur d'émission doit être intelligemment combiné à la puissance totale consommée de manière à assigner les charges - VMs et trafics - aux équipements les plus éco-énergétiques.

Enfin, le problème de placement des VMs et du routage du trafic, étant chacun NP-difficile, il en résulte que cette optimisation conjointe, combinée à l'intégration de l'impact du cooling, engendre un problème encore plus complexe à résoudre, d'où l'importance de développer une méthode de résolution approchée, permettant d'obtenir de bonnes solutions en un temps polynomial.

2.2 Réduction de l'énergie consommée

Dans le contexte d'un InterCloud, où les data centers sont alimentés par diverses sources d'énergie, renouvelables ou non, l'énergie totale dudit environnement ne saurait refléter fidèlement son impact écologique (Moghaddam *et al.*, 2011). Toutefois, dans le cas d'un data center alimenté par une énergie non verte, la réduction de son empreinte environnementale repose sur la minimisation de l'énergie consommée. Plusieurs mécanismes ont donc été développés, au fil du temps, pour pallier le problème de consommation énergétique dans les data centers (Parain *et al.*, 2000). Par conséquent, les principaux travaux réalisés dans le contexte de la minimisation de la puissance consommée seront présentés dans les lignes subséquentes.

2.2.1 Consommation énergétique dans le matériel

Pour répondre aux besoins croissants des usagers en termes de ressources informatiques, de récentes avancées technologiques ont permis d'intégrer plusieurs fonctionnalités et une puissance de calcul et de stockage de plus en plus importante aux équipements informatiques (Mittal, 2014). Toutefois, cette amélioration des caractéristiques des appareils est à l'origine d'une forte consommation énergétique pouvant engendrer des problèmes de fiabilité et nécessitant un système de refroidissement plus performant (Sentieys, 1997). Plusieurs approches ont donc été considérées afin de réduire l'énergie consommée par les systèmes informatiques.

2.2.1.1 Amélioration de l'architecture des équipements

Les premiers efforts ont particulièrement porté sur l'utilisation de nouveaux matériels et de techniques de fabrication permettant d'abaisser la tension d'utilisation des transistors (John, 2014), entraînant une consommation énergétique de 10 à 100 fois moins élevée (Zhai *et al.*, 2007). Toutefois, une tension d'opération trop proche de la valeur seuil engendre souvent des pertes de performances temporelles, augmentant à long terme l'énergie consommée. Cette perte de performance peut être contrecarrée de différentes façons : limitation de l'alimentation d'une composante aux blocs actifs (Su et Despain, 1995), réduction du nombre possible de changements d'états dans un circuit (Musoll et Cortadella, 1995), utilisation de composantes électroniques (Lorch et Smith, 1998), réduction de la taille de l'architecture matérielle (voir Hicks *et al.*, 1997) ou encore adaptation de la composante à la tâche à réaliser en utilisant des éléments reconfigurables (Pillement *et al.*, 2003).

2.2.1.2 Gestion des états de veille

Afin d'éliminer l'énergie en mode veille pouvant représenter jusqu'à 60% de la valeur maximale (Chen *et al.*, 2008), Isard *et al.* (2007) ont proposé des techniques pour désactiver des composantes de mémoire, tandis que Nedevschi *et al.* (2008) ont exploré des concepts similaires en vue de réduire la consommation des ressources réseau. Toutefois, pour éviter un long délai de réveil, des mécanismes supportant plusieurs états de fonctionnement, correspondant chacun à un niveau de consommation énergétique, ont été développés : détermination de la séquence des transitions d'états en considérant les contraintes de temps et de puissance (Li *et al.*, 2002) ; contrôle adaptatif des modes de fonctionnement basé sur l'historique d'arrivée des tâches et les délais d'exécution (Huang *et al.*, 2010) et interface de gestion permettant de spécifier les intervalles d'inactivité de certaines composantes (Hoeller Jr *et al.*, 2006).

2.2.1.3 Adaptation dynamique du voltage et de la fréquence ou Dynamic Voltage Frequency Scaling (DVFS)

La DVFS permet d'ajuster la fréquence de fonctionnement au nombre de tâches à effectuer (Navet et Gaujal, 2006), diminuant ainsi la tension d'alimentation et la puissance consommée. À cet effet, les travaux de Hua et Qu (2003) ont démontré que 3 à 4 niveaux de tension d'alimentation permettaient d'assurer une efficacité énergétique, au même titre que le système idéal à tension variable. Ils ont aussi développé des algorithmes qui établissent un compromis entre la réduction énergétique et les contraintes temporelles (voir Hua *et al.*, 2003). Deux approches implémentant la DVFS ont été proposées par Quan et Hu (2001) : le premier

détermine la plus petite fréquence à utiliser au cours de l'exécution d'une tâche donnée, alors que le second définit la séquence des fréquences à appliquer considérant les besoins courants de l'application en termes de performances. Les résultats ont démontré que le second algorithme permet d'économiser plus d'énergie que le premier. Kan *et al.* (2010) ont également présenté une technique de DVFS afin de réduire l'énergie dans les systèmes embarqués temps-réels.

2.2.2 Consommation des serveurs

Le mode d'utilisation des ressources affecte largement la quantité d'énergie consommée, en dépit des technologies utilisées (Lee et Zomaya, 2012). Avec le modèle IaaS, les clients soumettent leurs applications et il revient au fournisseur de déterminer l'emplacement idéal de ces dernières (Liu *et al.*, 2009). Dans le contexte d'une minimisation de l'énergie consommée, une assignation inefficace conduit souvent à une sous-utilisation de l'infrastructure physique et à un gaspillage énergétique. Afin de réduire l'énergie consommée, plusieurs travaux se sont intéressés au problème d'assignation des VMs aux serveurs adéquats, ces derniers représentant l'entité la plus énergivore dans un data center (John, 2014).

2.2.2.1 Serveurs homogènes

Grâce à la virtualisation et supposant une homogénéité des serveurs, la tâche principale consiste à déterminer comment grouper les VMs afin de réduire le nombre d'équipements actifs et désactiver les serveurs inutilisés. Aussi, lorsque les techniques de DVFS sont disponibles, il est également question, à cette phase, de déterminer la ou les fréquence(s) de fonctionnement idéale(s) de chaque serveur actif. La résolution de ce type de problème requiert, en général, des informations telles que le nombre de serveurs, leur capacité résiduelle et les caractéristiques des VMs en termes de ressources physiques nécessaires.

Les premiers travaux ont tenté de concilier les techniques de DVFS au processus de placement de la charge. Ainsi, pour une charge définie par la quantité de cycles CPU requise, et assumant une divisibilité de la charge ainsi qu'une homogénéité de l'infrastructure, Abdelsalam *et al.* (2009) ont développé une formule déterminant le nombre de serveurs physiques à garder actifs, de même que leur fréquence de fonctionnement. Xu *et al.* (2012) ont modélisé la charge par des VMs ayant des besoins en puissance de calcul et ont conçu un modèle multi-objectif, basé sur l'approche Paréto où le nombre de serveurs actifs, la puissance consommée par les VMs, incluant la fréquence de fonctionnement, et le gaspillage des ressources, ont été optimisés. Li *et al.* (2009) ont fait abstraction des techniques de DVFS, mais ont développé un mécanisme d'assignation des VMs à un nombre minimal de serveurs. L'originalité de ce travail se situe au niveau de l'approche utilisée. Partant du principe que dans une plateforme de Cloud, les VMs

arrivent et partent de manière dynamique, lorsqu'une nouvelle requête arrive, l'heuristique déplace les charges plus faibles et insère la nouvelle charge beaucoup plus lourde à leur place. Les charges déplacées sont, par la suite, relocalisées en suivant le même principe. La Figure 2.1 permet d'illustrer ce mécanisme. Feller *et al.* (2011) ont, pour leur part, modélisé la situation comme le problème de *Bin Packing* à dimensions multiples, où VMs et serveurs sont définis par un vecteur de ressources multi-dimensionnel. Ils ont représenté le problème à l'aide des techniques de programmation linéaire en nombres entiers, et ont proposé une adaptation de l'optimisation par Colonie de Fourmis pour résoudre les instances de grande taille. Les résultats ont démontré que l'algorithme proposé performe mieux comparé aux approches de résolution gloutonne, tels le First Fit Decreasing (FFD) ou le Best Fit (BF).

2.2.2.2 Serveurs hétérogènes

Dans le contexte d'immenses data centers, composés de plusieurs centaines de serveurs, l'homogénéité des équipements n'est pas de mise, de ce fait, d'autres techniques doivent être développées afin de réduire l'impact énergétique des serveurs. En ce sens, Kansal *et al.* (2010) ont introduit un modèle de consommation des serveurs qui comprend une composante fixe, la puissance en mode veille, et une composante qui varie en fonction de l'utilisation des ressources. Partant de ce modèle, Xie *et al.* (2013) ont développé un modèle de programmation linéaire minimisant l'énergie totale consommée par les serveurs, incluant l'énergie associée aux transitions d'états. Une approche gloutonne basée sur l'algorithme BF a aussi été proposée pour résoudre les problèmes de grande taille. Dupont *et al.* (2012) ont développé un modèle qui maximise la quantité d'énergie économisée lors d'un processus de reconfiguration des VMs, à l'échelle de plusieurs data centers, et ont utilisé une méthode basée sur la programmation par contraintes afin de déterminer la configuration optimale. Gao *et al.* (2013)

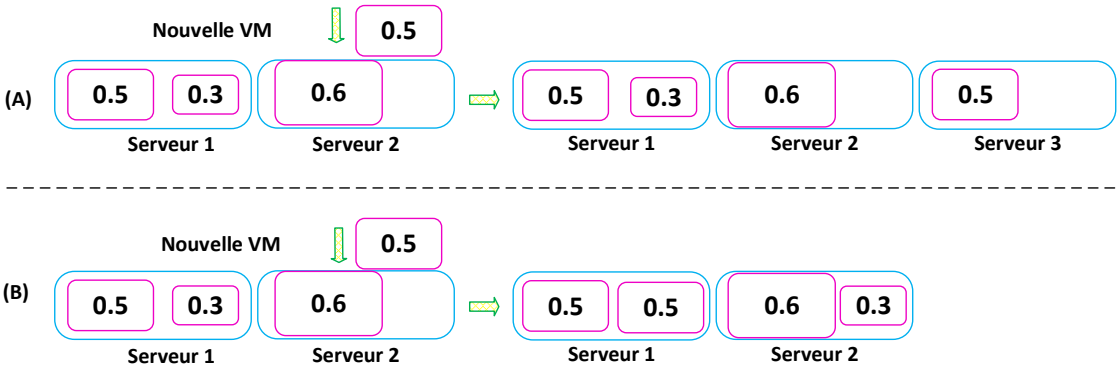


Figure 2.1 Processus de placement sans (A) et avec (B) relocalisation

ont proposé un modèle de placement optimal minimisant simultanément le gaspillage des ressources et l'énergie consommée. Une méthode basée sur l'approche Pareto a été proposée afin d'obtenir un ensemble de solutions non-dominées.

2.2.3 Impact du système de refroidissement

Un placement sous-optimal de la charge peut entraîner un accroissement de la température, qui peut, non seulement nuire au bon fonctionnement des équipements informatiques, mais peut également engendrer des coûts de refroidissement très élevés. De ce fait, plusieurs travaux ont proposé des solutions afin de résoudre le problème d'efficacité énergétique, lequelles visent à réduire la puissance consommée par le CRAC.

Patel *et al.* (2003) ont proposé, conjointement au processus de placement des VMs, un mécanisme de détection et de refroidissement des points chauds, basé sur des modules de modélisation, de mesures et de contrôles. Sharma *et al.* (2005) ont développé un système d'équilibrage des charges afin d'éviter un sous-refroidissement des racks et maintenir une uniformité au niveau de la température d'entrée, minimisant ainsi les points chauds. Ils ont développé 2 approches : la première redistribue la charge d'une rangée de serveurs de manière à équilibrer la température de sortie de cette dernière, alors que la seconde procède à une réallocation de la charge entre plusieurs rangées de racks. Moore *et al.* (2005) ont présenté deux (2) algorithmes visant à réduire l'énergie consommée par le CRAC. La première approche consiste à assigner les VMs de manière inversement proportionnelle à la température d'entrée des serveurs, alors que la seconde méthode permet de placer les charges afin de réduire la recirculation de chaleur. Les performances de leurs algorithmes ont été comparées à trois (3) méthodes d'équilibrage de la charge : la première vise à une répartition uniforme de la puissance CPU, la seconde assigne les charges aux serveurs inutilisés et de faible température d'entrée ; et la dernière répartit la puissance consommée afin d'éviter les points chauds.

Tang *et al.* (2006), pour leur part, ont proposé un modèle de prédiction de la distribution thermique à l'intérieur d'un data center. Ce modèle a été utilisé par Pakbaznia et Pedram (2009) et combiné au modèle d'évaluation du COP d'un système de cooling afin d'effectuer un placement optimal et minimiser la consommation des serveurs et du CRAC. Dans ce dernier travail, l'approche visant à augmenter la température fournie par le système de refroidissement afin de réduire la consommation énergétique de ce dernier, comme le suggère le rapport de la ASHRAE, a également été mise à profit. Toutefois, des études récentes ont mis en évidence le comportement dynamique des serveurs face à l'augmentation de la température. Plus précisément, Moss et Bean (2009) ont démontré expérimentalement que, face à une hausse de température, les ventilateurs des serveurs ajustent leur vitesse de rotation afin

d'accroître la quantité d'air frais pénétrant les serveurs, ce qui augmente, du même coup, leur consommation de puissance. Cette expérimentation avait pour but de mettre en évidence la nécessité de trouver une température de compromis afin de balancer l'accroissement de la puissance des ventilateurs et le gain de puissance du cooling à hautes températures. Suite à ces découvertes, Lee (2012) a présenté une étude de cas portant sur l'impact de la température d'entrée des serveurs sur l'efficacité énergétique à l'intérieur d'un data center.

2.2.4 Consommation des ressources réseau

Avec l'émergence des applications plus complexes nécessitant des interactions entre plusieurs VMs, et un trafic dans le réseau responsable d'environ 25% de la consommation énergétique d'un data center (voir Fang *et al.*, 2013a), réduire la puissance consommée par les commutateurs d'un data center est rapidement devenu une priorité.

En ce sens, Mahadevan *et al.* (2010) ont proposé un système qui réoriente le trafic afin d'obtenir une configuration où la consommation énergétique dans le réseau tend à être proportionnelle au trafic qui y est routé. Considérant une matrice de trafic et une topologie de data center, Shang *et al.* (2010) ont tenté de réduire l'empreinte énergétique des ressources réseau en minimisant le nombre de commutateurs et liens utilisés pour router le trafic inter-VMs, avec le moins de dégradation possible.

Meng *et al.* (2010) ont proposé une approche qui, considérant des serveurs identiques, des VMs homogènes, une matrice de trafic et le coût de communication entre chaque paire de serveurs, place les VMs afin que le trafic inter-VMs soit aligné avec leur distance de communication. Mann *et al.* (2011) ont présenté un cadre de planification des VMs basé sur de techniques de migration des VMs et de routage du trafic. Ils ont proposé une heuristique gloutonne, *VMFlow*, qui tient compte de la topologie du réseau, mais qui ne permet pas la consolidation de plusieurs VMs sur le même serveur. Biran *et al.* (2012) ont visé à placer les VMs de manière à satisfaire les demandes en termes de prédiction de trafic, et à assurer une certaine résilience. Deux (2) méthodes ont alors été proposées afin de résoudre ce problème : la première exploite les techniques de programmation en nombres entiers, et la seconde implémente une approche gloutonne. McGeer *et al.* (2010) ont formulé la minimisation de la puissance consommée dans le réseau comme un problème d'optimisation combinatoire et ont proposé une heuristique, valide uniquement pour la topologie Fat-Tree et basée sur la théorie des graphes. Fang *et al.* (2013b) ont proposé une nouvelle approche visant à optimiser simultanément le placement des VMs et le routage du trafic afin de réduire la puissance consommée dans le réseau. Ils ont proposé un algorithme de résolution à trois (3) étapes. La première phase regroupe les VMs afin de minimiser le trafic inter-groupes ; la seconde phase assigne les groupes de VMs

aux serveurs afin de réduire le trafic inter-racks ; et la troisième étape optimise le placement des VMs afin de réduire le nombre de commutateurs et liens utilisés.

2.2.5 Optimisation conjointe

À des fins de simplification, plusieurs travaux ont tenté d'aborder la question de consommation énergétique d'un data center en s'attaquant à un problème à la fois. Toutefois, le placement des VMs et le routage du trafic étant deux processus mutuellement dépendants, ils ne sauraient être traités séparément. De plus, comme présenté à la Section 2.2.3, l'allocation des VMs peut largement influencer la distribution thermique, laquelle peut, à son tour, avoir un impact sur la consommation des serveurs. Ainsi, dans le but de minimiser l'énergie consommée par un data center, une optimisation conjointe des différents facteurs influençant la consommation énergétique est donc de mise.

2.2.5.1 Nœuds de calcul et ressources réseau

Ainsi, certains auteurs ont abordé la question en considérant simultanément les nœuds de calcul et les ressources réseau. Wu *et al.* (2012) ont présenté un modèle qui optimise simultanément l'énergie des serveurs et du réseau de communication. Les VMs et les serveurs sont caractérisés par un vecteur de ressources multi-dimensionnel, le modèle linéaire est utilisé pour la consommation des serveurs et la puissance consommée par les équipements réseau est proportionnelle au trafic qui y transite. Ils ont développé une heuristique basée sur l'algorithme génétique afin de solutionner le problème de placement des VMs. Wu (2013) a également développé une approche de résolution basée sur l'adaptation du Recuit Simulé ou *Simulated Annealing (SA)*. Par la suite, les auteurs Tang et Pan (2014) ont conçu une version hybride de l'algorithme, soit l'algorithme mémétique afin d'améliorer les performances de l'algorithme génétique présentée par Wu *et al.* (2012). Quan *et al.* (2012) ont proposé des approches de migration de la charge afin d'optimiser le placement des VMs et le routage du trafic et réduire l'énergie consommée. La puissance des nœuds de calcul inclut la consommation des serveurs et une consommation fixe des ventilateurs associés. Une topologie full mesh a été considérée comme configuration d'interconnexion entre les serveurs et un modèle complexe de consommation des commutateurs a été présenté. Ils ont également proposé un algorithme glouton à deux (2) phases. Dong *et al.* (2013) ont utilisé des techniques de programmation linéaire en nombres entiers pour déterminer l'emplacement idéal des VMs afin de maximiser l'utilisation des ressources. Pour ce qui est des ressources réseau, ils ont défini un coût de communication proportionnel à la quantité de trafic circulant dans le réseau. Ils ont alors formulé le problème sous la forme d'une optimisation multi-objective visant à minimiser

l'impact énergétique des VMs et de leurs trafics, et ont proposé une approche combinant une méthode de regroupement, ou *clustering*, à un algorithme glouton. Vu et Hwang (2014) ont développé un algorithme à trois (3) phases pour solutionner un tel problème. La première phase vérifie l'état des serveurs ; la seconde étape sélectionne les VMs à migrer ; et la dernière phase trouve la destination des VMs à migrer en considérant l'énergie et le trafic dans le réseau. Fang *et al.* (2013a) ont présenté une approche de placement et de migration des VMs où leur modèle intègre la puissance des serveurs, celle des ressources réseau et des coûts de migration. Ils proposent une adaptation de trois (3) algorithmes gloutons, soit Best Fit, First Fit et Worst Fit, pour résoudre les problèmes de grande taille. Larumbe et Sanso (2012) ont proposé une approche multi-objective dans laquelle le modèle de consommation énergétique considère simultanément l'énergie des serveurs, celle des ressources réseau et un PUE fixe. Le modèle proposé est basé sur la programmation linéaire en nombres entiers et est résolu à l'aide d'AMPL/CPLEX. Ces auteurs ont également proposé une adaptation de l'heuristique de la Recherche Tabou afin de résoudre les problèmes de grande taille et ont comparé les performances de cette méthode avec celles d'une approche de résolution gloutonne (voir Larumbe et Sanso, 2013). Ils ont également été parmi les rares à évaluer l'énergie consommée à l'échelle de plusieurs data centers.

2.2.5.2 Nœuds de calcul et système de refroidissement

D'autres auteurs, pour leur part, ont tenté de réduire l'énergie consommée par un data center en minimisant l'impact des serveurs et du cooling. En utilisant les chaînes de Markov, Parolini *et al.* (2008) ont proposé un mécanisme de placement des VMs réduisant simultanément la consommation des nœuds de calcul et l'énergie du système de refroidissement. Xu et Fortes (2010) ont proposé une approche de placement des VMs basée sur une optimisation multi-objective, visant à réduire le gaspillage des ressources, la puissance consommée et le coût de dissipation thermique. Ils ont développé un modèle de consommation des serveurs basé uniquement sur l'utilisation CPU, et l'impact du cooling est contrôlé en minimisant la température de sortie des serveurs. Une adaptation de l'algorithme génétique combinée à la logique floue a été développée afin de déterminer des solutions de compromis entre les différents objectifs. Pakbaznia et Pedram (2009) ont proposé un modèle d'optimisation de l'énergie totale dans un data center. Ce modèle, basé sur la programmation linéaire en nombres entiers, permet d'effectuer un placement optimal des VMs en minimisant la puissance consommée par les serveurs et celle dissipée par le cooling. Ils ont utilisé le modèle de dissipation thermique proposé par Tang *et al.* (2006) et ont maximisé l'efficacité énergétique du système de refroidissement en augmentant la température interne du data center, comme suggéré par une étude de la ASHRAE (voir ASHRAE, 2011).

2.3 Énergie et performances

Lorsque plusieurs VMs sont consolidées sur le même serveur, si elles sont incompatibles, leurs performances se dégradent, entraînant une plus longue durée d'exécution des tâches et un surplus d'énergie (Sharifi *et al.*, 2012). Dès lors, il importe de s'interroger sur la nature et le nombre de VMs à consolider, avant même de procéder à cette consolidation. Ainsi, afin d'éviter toute forme de consolidation aveugle, plusieurs travaux ont tenté de trouver le meilleur compromis entre la minimisation de la puissance électrique et le respect des critères de performance des applications hébergées.

Bobroff *et al.* (2007) ont proposé un mécanisme qui détermine le nombre de serveurs nécessaires afin d'assurer un niveau de QdS pour une charge donnée. Partant de l'historique des données, leur algorithme prédit les demandes futures et tente de relocaliser les VMs afin de minimiser le nombre de serveurs actifs. Verma *et al.* (2008a) ont défini trois (3) modèles d'optimisation permettant de considérer le problème d'équilibre entre la consommation énergétique et le niveau de performance des applications : soit l'optimisation conjointe des performances et de l'énergie consommée ; ou encore la minimisation de l'énergie consommée et des coûts de migration, sous la contrainte d'une garantie de performance ; ou encore la maximisation du bénéfice net lié aux performances en considérant une consommation énergétique fixe des serveurs. Ces auteurs ont adapté leurs travaux afin de considérer le placement des applications nécessitant des calculs haute performance ou *High Performance Computing*, où les performances sont liées, à la fois, à l'utilisation CPU et au nombre de VMs (voir Verma *et al.*, 2008b). Srikantaiah *et al.* (2008) ont développé une technique qui, en considérant des seuils d'utilisation prédéfinis pour chaque serveur, amortit l'énergie en mode veille, tout en limitant la consommation due aux conflits internes.

Cardosa *et al.* (2009) ont visé à maximiser l'utilité en déterminant la quantité de ressources à assigner à chaque VM, et à déterminer un schéma de placement idéal permettant de minimiser le nombre de serveurs actifs. Ils ont développé un algorithme qui dimensionne une VM en considérant la quantité de ressources disponibles, la puissance consommée et le niveau de priorité de la tâche ou l'utilité qui lui est associée. Les auteurs Li *et al.* (2009) ont, quant à eux, implémenté une approche de sur-approvisionnement afin de considérer le phénomène de variation des charges et de dégradation de performance. Des mécanismes de redimensionnement ou de migration sont également considérés afin de réduire l'énergie consommée ou dans le but d'éviter des dégradations de performance. Van *et al.* (2010) ont, pour leur part, considéré deux sous-problèmes distincts, mais inter-reliés : le problème d'approvisionnement qui maximise le nombre de VMs assignées aux applications afin d'assurer leur bon fonctionnement ; et le problème de placement des charges qui minimise la consommation énergétique.

Kusic *et al.* (2009) ont proposé un modèle de maximisation du profit, lequel intègre également des pertes de revenu dues aux transitions d'états d'un serveur ou aux délais d'allumage d'une VM. Un contrôleur qui détermine le nombre de VMs, la quantité de ressources à assigner aux applications, de même que l'emplacement idéal de ces dernières, a également été développé. Mazzucco *et al.* (2010) ont aussi proposé un modèle de maximisation du profit en utilisant les notions de files d'attente afin de modéliser le comportement général du système. Deux méthodes de placement des VMs, soit une heuristique qui s'adapte à la variation des charges, et une approche qui prédit les demandes futures, ont également été développées.

Beloglazov *et al.* (2012) ont mis en place un algorithme de placement des VMs qui se divise en 2 phases : l'assignation d'une nouvelle requête à un serveur, en utilisant l'approche du *Modified Best Fit Decreasing (MBFD)* ; et la reconfiguration des VMs. Cette dernière comprend elle aussi 2 phases, soit la sélection des VMs à migrer, et le placement des VMs migrées à l'aide du MBFD. Sharifi *et al.* (2012) ont introduit un coefficient de consolidation ou *Consolidation Fitness (CF)* qui mesure l'aptitude d'un algorithme à réaliser un placement efficace en vue de réduire la consommation tout en maintenant une performance adéquate. Cette métrique se définit comme le rapport entre le taux de violations des performances, dues à la consolidation, sur le pourcentage de gain en énergie, résultant de ce même processus de consolidation. Un algorithme de placement minimisant des métriques telles que la puissance et les violations de performances, évaluées à l'aide du coefficient CF, a aussi été développé.

Dans les travaux réalisés par Kim *et al.* (2013), l'algorithme proposé utilise un modèle d'interférence qui détermine l'impact mutuel des VMs co-localisées, et où les auteurs évitent de combiner les charges hautement sensibles avec des VMs dont l'impact en termes de dégradation de performance induite est très élevé. Kord et Haghighi (2013) ont, quant à eux, mis en place une technique de consolidation des VMs basée sur la méthode de coefficient de corrélation minimal, et ont utilisé une approche de résolution considérant les outils d'aide à la décision multicritères, afin de trouver le meilleur compromis entre l'énergie consommée et les performances des applications.

2.4 Minimisation de l'impact environnemental

Afin de limiter l'empreinte carbone des data centers, plusieurs auteurs ont cherché à déterminer des mécanismes permettant de minimiser la quantité de GES rejetés par ces derniers.

En ce sens, Moghaddam *et al.* (2011) ont introduit une formulation mathématique définissant l'empreinte carbone d'un data center comme le produit entre le facteur d'émission et l'énergie consommée. Par la suite, ils ont proposé une approche basée sur l'algorithme génétique afin

d'optimiser le placement des VMs à l'échelle de plusieurs data centers et minimiser leur empreinte carbone. Ils ont aussi développé un modèle plus précis de consommation énergétique d'une VM, lequel a été, par la suite, intégré à une version améliorée de l'algorithme génétique, le *Multi-Level Grouping Genetic Algorithm (MLGGA)*, afin de minimiser l'empreinte carbone d'un ensemble de data centers (voir Moghaddam *et al.*, 2012). Liu *et al.* (2011) ont proposé un algorithme d'assignation des VMs à des data centers, le *Geographical Load Balancing (GLB)*, permettant de tirer profit des énergies renouvelables, particulièrement des énergies éoliennes et solaires. Une approche similaire a été considérée par Van Heddeghem *et al.* (2012), qui ont introduit un modèle basé sur la probabilité de la fonction de masse de la loi binomiale afin d'évaluer l'empreinte carbone d'un environnement InterCloud. Ce modèle a été intégré au mécanisme basé sur l'approche *Follow The Sun/Follow The Wind* (voir Figure 2.2), qui consiste à migrer les charges de manière dynamique en considérant la disponibilité des sources d'énergie renouvelables. Certains auteurs (voir Larumbe et Sanso, 2012, 2013) ont également considéré la minimisation de l'empreinte carbone dans leur modèle d'optimisation multi-objective. Ils ont utilisé une approche similaire à Moghaddam *et al.* (2011), où la quantité de GES se calcule en considérant la consommation énergétique des data centers et les sources d'énergie alimentant ces derniers.

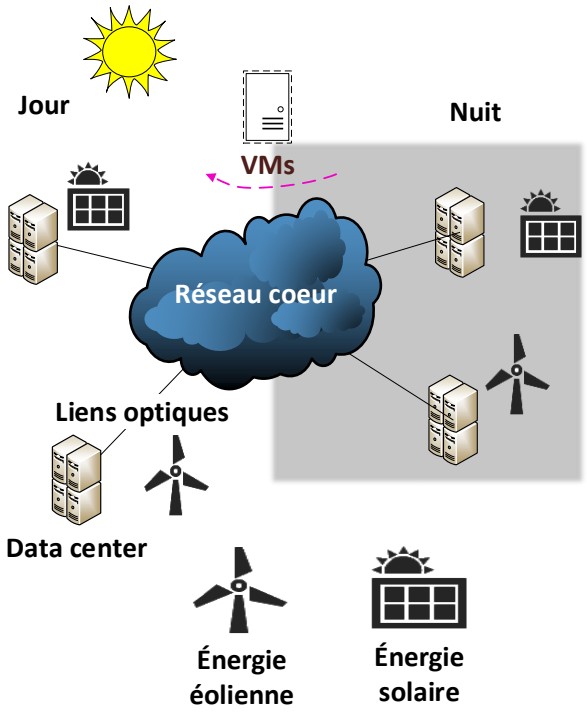


Figure 2.2 Processus de relocalisation des VMs selon le principe FTSFTW

Garg *et al.* (2011b) ont mis en place une nouvelle architecture et une politique de placement des VMs, le *Carbon Efficient Green Policy (CEGP)* afin de réduire l'empreinte carbone dans un environnement de clouds fédérés. L'algorithme proposé liste les VMs en considérant les contraintes de temps ou *Earliest Deadline First (EDF)*, afin d'éviter toute violation de QdS et classe les data centers par ordre croissant de leur facteur d'émission. Par la suite, un algorithme glouton assigne chaque VM au data center induisant la plus faible quantité de CO_2 en considérant le facteur d'émission dudit data center, son PUE et la consommation énergétique associée à la VM. Une approche similaire a été présentée par Khosravi *et al.* (2013) qui ont proposé une adaptation du BF, le *Energy and Carbon-Efficient*, plaçant chaque VM sur le serveur dont l'empreinte carbone est minimale. Wadhwa et Verma (2014) ont utilisé le modèle mathématique développé par Khosravi *et al.* (2013) et ont proposé un algorithme à deux phases : soit le placement initial et l'optimisation de l'énergie via des mécanismes de migration. À l'arrivée d'une nouvelle VM, l'algorithme place cette dernière sur un serveur quelconque du data center ayant le plus faible facteur d'émission, de manière à réduire d'emblée l'empreinte carbone due à l'hébergement de la VM. À la seconde phase, via des processus de migration à l'échelle des serveurs d'un unique data center, le placement des VMs est optimisé afin de réduire la puissance consommée par ledit data center.

2.5 Méthodes de placement utilisées

Les méthodes utilisées pour résoudre le problème de placement des VMs et du trafic associé dans les data centers peuvent être classées en trois (3) catégories : les méthodes exactes, les méthodes approchées et les méthodes hybrides.

2.5.1 Méthodes exactes

Les algorithmes complets permettent de répondre, de manière exacte, à un problème d'optimisation. Ces méthodes donnent une garantie de trouver la solution optimale. Toutefois, lorsque la taille des problèmes augmente, particulièrement dans le cas des problèmes NP-difficiles, le temps de traitement nécessaire à l'obtention de la solution optimale croît exponentiellement. Dans la littérature, on recense plusieurs algorithmes exacts.

2.5.1.1 Énumération

L'approche la plus simple est la méthode d'énumération qui consiste à déterminer toutes les configurations possibles, à en évaluer le coût et à choisir la meilleure solution. Toutefois, en raison des nombreuses mises à jour nécessaires et surtout des problèmes de grande taille,

il devient très difficile, voire même impossible, d'évaluer l'éventail de toutes les possibilités à la main, particulièrement lorsque le nombre de contraintes à vérifier est très élevé. Ainsi, des approches automatiques, comme la programmation par contraintes ou les méthodes de programmation linéaire, ont été privilégiées au détriment de la résolution manuelle.

2.5.1.2 Programmation par contraintes

La méthode de programmation par contraintes revient à modéliser le processus par des variables appartenant à un espace fini et liés par des contraintes mathématiques (Rousseau et Pesant, 2005). La recherche de solutions s'effectue alors par le biais de déductions logiques, où des algorithmes de propagation de contraintes font abstraction des valeurs de variables engendrant des solutions non faisables. Des mécanismes plus poussés, comme la recherche arborescente, sont également utilisés pour améliorer la recherche. Les méthodes de programmation par contraintes ont été utilisées pour mettre en place un mécanisme de placement dynamique et de migration des applications (Hermenier *et al.*, 2009; Anderson *et al.*, 2010; Hermenier *et al.*, 2011) ; pour équilibrer la répartition des applications à l'échelle de plusieurs hôtes physiques (Fukunaga, 2009) ; pour traiter le problème d'approvisionnement et de placement des VMs (Van *et al.*, 2010), ou encore afin de minimiser l'énergie consommée tout en considérant les contraintes de performances des applications hébergées (Dupont *et al.*, 2012).

2.5.1.3 Programmation mathématique

Ces techniques de programmation mathématique permettent de modéliser le problème combinatoire sous la forme d'un système d'équations à résoudre, lesquelles représentent les contraintes du problème, dans le but d'optimiser un objectif particulier. Les méthodes de programmation linéaire sont utilisées lorsque la fonction objectif et toutes les équations sont linéaires, dans le cas contraire, on parle de programmation non linéaire. De manière générale, la programmation mathématique fait intervenir des variables continues, toutefois, il existe des variantes à cette technique de résolution, telles la programmation binaire, en nombres entiers, ou la programmation mixte, selon la nature des variables en question. Plusieurs auteurs ont utilisé les méthodes de programmation mathématique afin de résoudre le problème de placement des VMs. Bichler *et al.* (2006) et Speitkamp et Bichler (2010) ont modélisé le processus de consolidation des applications à l'aide de techniques de programmation linéaire, visant à réduire le nombre de serveurs actifs. Chaisiri *et al.* (2009) ont développé une approche d'optimisation orientée utilisateurs, *Optimal Virtual Machine Placement (OVMP)*, leur permettant de réduire au maximum les dépenses d'hébergement de leurs VMs. Une approche de programmation linéaire en nombres entiers a été également utilisée par les auteurs

Larumbe et Sanso (2012) afin de réduire le coût d'hébergement d'un ensemble de VMs à l'échelle de plusieurs data centers géographiquement distribués. Ces auteurs ont défini un modèle multi-objectif afin d'optimiser le délai, l'énergie consommée et l'empreinte carbone.

2.5.2 Méthodes approchées

Le processus de placement des VMs et celui du routage du trafic sont deux problèmes NP-difficiles, qui lorsque combinés, engendrent un problème encore plus difficile. En considérant un cas simple de placement des VMs, où seules les contraintes en termes de capacité physique des équipements sont prises en compte, le problème demeure difficile, car la taille de l'espace de recherche est le nombre de VMs élevé à une puissance correspondant au nombre de serveurs. Ainsi, pour les grandes instances de problèmes, le temps nécessaire à l'évaluation de toutes les possibilités peut conduire à une explosion combinatoire. Ainsi, des méthodes approchées, permettant de trouver de bonnes solutions, en un temps raisonnable, ont été développées et utilisées pour résoudre le problème de placement des VMs et du routage du trafic.

2.5.2.1 Méthodes de construction progressive

Ces approches permettent de construire la solution au fur et à mesure, en se basant sur des règles d'assignation prédéfinies, sans pour autant revoir les décisions prises aux étapes précédentes. Ces méthodes regroupent, entre autres, les algorithmes gloutons qui consistent à trouver, pour chaque VM, le meilleur data center et serveur hôtes. Elles recherchent, par conséquent, pour chaque VM, l'optimum local. Toutefois, en dépit de leur rapidité, ces méthodes n'aboutissent à la solution optimale qu'en cas d'homogénéité de l'infrastructure sous-jacente. Dans le cas contraire, la recherche d'un optimum local (meilleur serveur hôte) pour chaque VM ne saurait garantir l'optimalité de la solution. Les méthodes gloutonnes ont été utilisées afin de résoudre le problème de placement des VMs en considérant l'énergie consommée et la dégradation de performance des applications (Beloglazov *et al.*, 2012; Quan *et al.*, 2012). Dong *et al.* (2013) ont également utilisé un algorithme glouton combinant le *Minimum Cut* et le Best Fit Decreasing (BFD) afin de maximiser l'utilisation des ressources dans le but ultime de réduire l'énergie consommée. Le *Minimum Cut* permet de regrouper au niveau d'un même commutateur, les VMs avec un large trafic inter-VMs, afin de réduire le nombre de commutateurs actifs. Puis considérant la configuration résultant de la première phase, le BFD optimise le placement des VMs afin de réduire le nombre de serveurs actifs. En raison de leur simplicité d'implémentation et de leur rapidité à trouver une solution, ces méthodes gloutonnes sont souvent utilisées pour déterminer un schéma de placement initial qui sera,

par la suite, amélioré par des mécanismes plus complexes, telles les heuristiques.

2.5.2.2 Métaheuristiques

Une métaheuristique est une approche qui, par le biais de mécanismes aléatoires et itératifs, explore l'espace de recherche, tout en définissant des stratégies d'apprentissage du milieu, dans le but de déterminer des solutions de qualité. Une métaheuristique débute avec une solution initiale et tente, de manière itérative, d'améliorer le coût de cette dernière en visitant les configurations voisines. Une adaptation de la métaheuristique choisie est de mise afin de solutionner le problème d'intérêt, dont les enjeux vont de la simple approche de représentation des solutions, à la définition de mécanismes d'exploration de l'espace de recherche, ou même au réglage des paramètres de simulations. En général, on distingue les méthodes de trajectoire ou de recherche locale, qui sont des algorithmes travaillant avec une unique solution, des métaheuristiques à base de population.

Recherche locale Les techniques de recherche locale explorent l'espace de recherche en générant, de manière itérative, une nouvelle solution dans le voisinage de la solution actuelle, par le biais de mouvements adaptés au problème en question. La solution initiale est souvent construite à l'aide d'un algorithme glouton et plusieurs critères d'arrêt, comme le nombre d'itérations, le temps limite imposé, peuvent être utilisés pour arrêter le processus.

La stratégie de recherche locale la plus simple est la descente. Partant d'une solution initiale, à chaque itération, l'algorithme choisit la meilleure solution, dans le voisinage de la configuration actuelle. L'exploration s'arrête si, à une itération donnée, il n'existe aucun voisin améliorant le coût de la solution actuelle. Quoique rapide, le principal inconvénient de cette méthode de descente est son arrêt prématuré au premier minimum local trouvé, lequel minimum peut être assez loin de l'optimum global, en termes de coût de la solution, comme l'illustre la Figure 2.3. Pour améliorer les performances de cet algorithme, une stratégie consiste à perturber la solution actuelle représentant le minimum local de manière à en modifier certains attributs. Ceci permet de déclencher le processus de descente sur une nouvelle configuration. Dans certains cas, les perturbations peuvent être répétées un certain nombre de fois, ce qui permet d'augmenter les chances d'explorer un plus grand nombre de solutions. C'est la Recherche Locale Itérée ou ILS. Il est à noter que la force des perturbations réalisées sur l'optimum local, à la fin d'un processus de descente, doit être convenablement contrôlée, de manière à éviter qu'on ne s'éloigne trop de cette bonne solution. Les auteurs Leivadeas *et al.* (2013) ont proposé une adaptation du ILS pour partitionner les demandes des utilisateurs à l'échelle de plusieurs data centers, dans le but de minimiser le taux de rejets

des VMs.

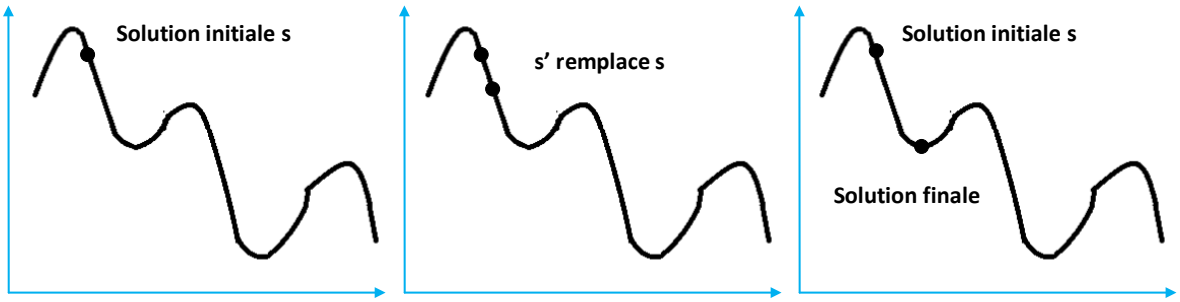


Figure 2.3 Évolution d'une solution dans l'algorithme de descente

Il existe également des métaheuristiques qui dérivent du domaine de la physique, comme le Recuit Simulé ou SA. Contrairement aux méthodes de descente, le Recuit Simulé permet de s'échapper des optima locaux. Il repose sur des techniques utilisées en métallurgie et sur des travaux décrivant l'évolution d'un système thermodynamique. L'algorithme débute avec une solution initiale s_0 à laquelle s'associe une énergie E_0. À chaque itération, une solution est choisie au hasard dans le voisinage de la solution actuelle. Ce voisin est automatiquement accepté si l'énergie qui lui est associée est inférieure à celle de la configuration courante. Dans le cas contraire, cette solution voisine n'est acceptée qu'avec une certaine probabilité. Une adaptation du SA a été proposée par Wu (2013) afin de réduire l'énergie consommée par les serveurs et les ressources réseau.

L'algorithme de Recherche Tabou ou *Tabu Search (TS)* présente une structure similaire au SA, mais permet également d'éviter de cycler autour d'une bonne solution. Partant d'une solution initiale, s_0, à chaque itération, l'algorithme choisit toujours la meilleure configuration, s_{i+1} dans le voisinage de la solution actuelle, s_i, que cette configuration voisine améliore le coût de s_i ou pas. Toutefois, dans l'éventualité que s_i soit également la meilleure solution dans le voisinage de s_{i+1}, l'algorithme aura tendance à cycler autour de ces deux solutions. Pour remédier à ce problème, l'algorithme introduit un mécanisme de mémoire à court terme, soit la liste tabou, lui permettant de garder un historique des récentes solutions, de façon à ce que ces dernières ne soient pas sélectionnées pour un certain nombre k d'itérations. Toutefois, l'existence d'un critère d'aspiration force l'algorithme à parfois choisir une solution, même lorsque cette dernière est taboue. Un exemple de critère d'aspiration consiste à sélectionner une configuration taboue si cette dernière est meilleure que toutes les configurations évaluées jusque-là. Outre la liste taboue, des mécanismes de mémoire à moyen (Intensifica-

tion) et long (Diversification) termes peuvent être également implémentés afin d'optimiser les performances de la méthode. Alors que l'Intensification approfondit la recherche autour des meilleures solutions trouvées lors de la Recherche locale, la Diversification relance la recherche dans une zone totalement inexplorée, ce qui permet d'élargir l'éventail des solutions possibles. Larumbe et Sanso (2013) ont proposé un algorithme basé sur la Recherche Tabou afin de solutionner le problème de placement des VMs dans un environnement InterCloud.

Métaheuristiques à base de population Les algorithmes à base de population, à l'inverse des méthodes de recherche locale, font évoluer un ensemble de solutions simultanément. Ces méthodes sont caractérisées par plusieurs itérations d'une phase de coopération, suivie d'une étape d'adaptation. À l'étape de coopération, les solutions sont analysées et combinées entre elles afin de produire de nouvelles solution-enfants héritant des meilleures caractéristiques des configurations parentes. À la phase d'adaptation, chaque configuration évolue de manière indépendante. À l'instar des stratégies de recherche locale, plusieurs critères d'arrêt peuvent être définis pour mettre fin à l'exploration de l'espace de recherche.

L'optimisation par colonies de fourmis est un algorithme à base de population s'inspirant de l'éthologie. La conception de cette métaheuristique repose sur le fait que les individus (ou solutions) d'une population se partagent le savoir, ce qui guide leurs options futures et orientent leurs compatriotes dans les choix à effectuer. Plus particulièrement, dans leur quête de nourriture, les fourmis éclaireuses, grâce aux phéromones dégagées sur leur passage, parviennent à indiquer à leurs congénères le plus court chemin reliant le nid à la source de nourriture, lequel chemin se distingue par la forte concentration de phéromones qui y est déposée. Un algorithme basé sur l'optimisation par colonies de fourmis a été proposé par Feller *et al.* (2011) afin d'assigner les VMs aux serveurs dans le but de réduire le nombre d'équipements actifs et minimiser l'énergie consommée. Cette méthode de résolution a également été utilisée par Gao *et al.* (2013) afin de réduire l'énergie consommée tout en maximisant l'utilisation des ressources.

L'algorithme génétique est un exemple d'algorithme à base de population, et dont le principe de fonctionnement repose sur la sélection naturelle et la génétique. L'algorithme débute avec une population initiale de solutions potentielles (individus ou chromosomes). À chaque itération (ou génération), des opérateurs évolutionnaires simples, tels que la sélection, le croisement, la mutation, sont appliqués à la population actuelle afin de produire la génération suivante. Au cours de ce processus, certains individus se reproduisent, d'autres disparaissent, et les mieux adaptés parviennent à survivre au fil des générations. Une fois le critère d'arrêt atteint, par exemple le nombre de générations (ou le nombre d'itérations), la meilleure solution trouvée, qui représente le meilleur individu de la population, est renvoyée et l'al-

gorithme s'arrête. Mi *et al.* (2010) ont proposé une adaptation de l'algorithme génétique à des fins de migrations des VMs. Xu et Fortes (2010) ont également développé une approche basée sur l'algorithme génétique dans le but de minimiser le gaspillage de ressources, l'énergie consommée et la dissipation thermique. Wu *et al.* (2012) ont aussi utilisé ces techniques afin de minimiser l'énergie consommée par les serveurs et les ressources réseau.

2.5.3 Méthodes hybrides

Certains auteurs, ayant récemment reconnu les potentialités pouvant résulter de la combinaison des méthodes exactes et des métaheuristiques, ont tenté de développer de nouvelles méthodes de résolution permettant d'exploiter les avantages découlant de l'hybridation de ces approches, dites classiques.

Plusieurs techniques d'hybridation, telles la coopération et l'intégration, peuvent être considérées. Alors que dans l'approche de coopération, les algorithmes, exacts ou approchés, sont exécutés de manière séquentielle, pour la seconde technique, une des méthodes est intégrée dans l'autre.

Ainsi, Pakbaznia et Pedram (2009) ont développé un modèle de programmation en nombres entiers et ont utilisé une approche hybride afin de minimiser l'énergie consommée par les serveurs et le système de refroidissement d'un data center. Ils ont développé une méthode coopérative où le problème est d'abord résolu en faisant intervenir les techniques de programmation linéaire à la première phase, et dont les résultats sont, par la suite, utilisés dans un algorithme afin de déterminer l'ensemble des solutions faisables. Speitkamp et Bichler (2010) ont également utilisé une approche coopérative, mais où une heuristique basée sur les techniques de relaxation linéaire a été développée à la première phase, suivie de la résolution du problème par les approches de programmation linéaire en nombres entiers. Cette méthode a été utilisée afin de réduire le temps de calcul à la seconde phase.

Tang et Pan (2014), pour leur part, ont proposé une version hybride de l'algorithme génétique afin d'améliorer les performances de l'approche présentée par Wu *et al.* (2012) visant à solutionner le problème de minimisation de l'énergie dans un data center. Ils ont utilisé une approche d'intégration, où des procédures permettant de contrer la non-faisabilité de certaines solutions, combinées à des mécanismes d'optimisation locale, ont été intégrées afin de maximiser l'exploration de l'espace de recherche. Une métaheuristique hybride a aussi été développée par Bilal *et al.* (2014) afin de résoudre un problème de partitionnement de graphes. En effet, les auteurs ont proposé une approche intégrative, *Iterated Tabu Search (ITS)*, exploitant les avantages de deux métaheuristiques connues, Iterated Local Search et Tabu Search. Plus particulièrement, la conception de leur méthode est basée sur l'implémentation du ILS

où la recherche locale, après chaque perturbation, est effectuée à l'aide des techniques de Recherche Tabou (TS). Les résultats ont pu démontrer que ITS donne toujours de meilleurs résultats, comparé aux différentes méthodes prises séparément.

2.6 Analyse des travaux présentés dans la littérature

Comme nous avons pu le remarquer, plusieurs auteurs se sont penchés sur le problème de planification des applications dans les environnements du Cloud Computing afin de réduire leur consommation énergétique ou encore, leur impact environnemental. Cependant, une rapide analyse nous porte à croire que certaines avenues de recherche demeurent encore inexplorées.

La littérature actuelle sur la réduction énergétique à l'échelle d'un unique data center est très fournie. En effet, conscients que les serveurs représentent une grande part, en ce qui a trait à l'énergie consommée, plusieurs auteurs ont mis en œuvre des méthodes afin de réduire le nombre de serveurs actifs nécessaires à l'hébergement des applications. Toutefois, l'hétérogénéité des équipements remet rapidement en question cette manière de procéder. Naissent alors des techniques exploitant le profil énergétique des équipements, conjointement avec leur capacité, en termes de ressources physiques, afin de minimiser leur consommation énergétique. Cette dernière approche atteint ses limites lorsque vient le temps d'héberger des applications plus complexes qui nécessitent l'interaction entre plusieurs VMs. Un placement efficace des VMs combiné à un routage du trafic visant à optimiser conjointement la puissance des serveurs et celle des ressources réseau est donc de mise afin de réduire l'impact énergétique d'un data center.

Par ailleurs, bien que la virtualisation et le processus de consolidation des applications semblent répondre au problème de gaspillage des ressources et de consommation énergétique, ils sont souvent à l'origine de contentieux lorsque vient le temps, pour les applications co-hébergées, d'accéder aux ressources physiques. Ceci résulte souvent en un temps d'exécution rallongé et une dégradation au niveau des performances. Ainsi, plusieurs auteurs ont proposé diverses méthodes de placement des VMs permettant de consolider efficacement la charge tout en s'assurant d'une certaine isolation des performances. Toutefois, la plupart des travaux de recherche se sont particulièrement concentrés sur les caractéristiques intrinsèques des applications lors du processus de consolidation, et ont négligé d'autres contraintes de co-localisation en rapport avec la sécurité et la redondance, ce qui peut grandement aller à l'encontre des exigences et attentes des utilisateurs.

L'enjeu entourant le problème de réduction énergétique d'un ensemble de data centers est

encore plus de taille. En effet, bien qu'il soit possible d'adapter le modèle de consommation énergétique d'un unique data center, il importe, à l'échelle d'un InterCloud, non seulement de conjuguer avec la nature hétérogène des équipements, mais également de s'intéresser à l'indicateur d'efficacité énergétique, ou PUE, qui représente le rapport entre l'énergie totale consommée par un data center et celle dépensée uniquement par le fonctionnement des équipements informatiques. Cet indicateur fait référence, en particulier, à l'impact énergétique du système de refroidissement et est souvent différent d'un data center à l'autre. En se basant sur les travaux recensés dans la littérature et qui visent à résoudre le problème de consommation énergétique dans un InterCloud, très peu d'auteurs ont intégré ce paramètre lorsque vient le temps de déterminer l'emplacement idéal d'une VM donnée, ce qui résulte en une configuration sous-optimale en termes d'énergie consommée. De plus, une analyse plus poussée a également indiqué que ce PUE tend couramment vers des valeurs très élevées, alors qu'il serait possible de le ramener le plus que possible à l'unité. En ce sens, plusieurs approches ont été développées afin de réduire l'énergie associée au cooling. De manière générale, elles visent soit à équilibrer les charges afin d'éviter des points chauds, ou encore à minimiser la recirculation de la chaleur. Une approche plus directe de réduction de l'impact du cooling consiste à augmenter la température interne d'un data center. Toutefois, ces mécanismes ne tiennent pas compte du comportement dynamique des ventilateurs des serveurs face à un accroissement de leur température d'entrée. Ainsi, un processus de placement faisant abstraction de ce type de phénomène peut, à l'inverse, résulter en une configuration où l'énergie consommée est beaucoup plus élevée qu'attendue.

Il existe, au sein de la littérature, quelques travaux portant sur l'optimisation de l'empreinte carbone d'un ensemble de data centers géographiquement distribués. Bien que la méthode intuitive, consistant à déporter certaines applications gourmandes en énergie électrique vers des data centers alimentés par des énergies vertes, semble la plus prometteuse, elle ne résulte pas nécessairement en une configuration optimale, en raison des équipements de caractéristiques différentes. De plus, afin de solutionner le problème de réduction des GES, l'approche de réduction énergétique, utilisée pour minimiser l'énergie d'un InterCloud ne saurait s'appliquer telle quelle, car, considérant un environnement où les data centers sont alimentés par différentes sources d'énergie renouvelables, la consommation énergétique dudit environnement ne saurait nécessairement refléter son impact environnemental. De ce fait, plusieurs techniques d'optimisation de l'empreinte carbone ont été proposées où, en général, la puissance consommée est combinée au facteur d'émission des sources d'énergie alimentant les data centers. Toutefois, pour l'ensemble de ces travaux visant à réduire l'impact environnemental, seul le placement des VMs sur les serveurs est optimisé. Sont alors laissés pour compte l'optimisation de l'énergie consommée par le CRAC et les ressources réseau. De plus, aucune technique de

consolidation efficace visant à réduire la dégradation des performances n'a été considérée.

Dans cette thèse, nous nous donnons donc pour tâche d'intégrer tous les éléments mentionnés ci-dessus, afin de proposer un modèle complet d'optimisation de l'empreinte carbone dans un environnement InterCloud. La mise en œuvre d'un tel modèle global est un travail assez complexe, mais aura le mérite de fournir d'excellentes solutions. En effet, cette approche permettra non seulement d'intégrer les différents facteurs influençant l'impact environnemental des data centers, mais facilitera également la préservation des principales interactions entre ces entités. Plus particulièrement, le modèle proposé visera à déterminer un schéma d'allocation statique d'un ensemble d'applications à l'échelle de plusieurs data centers. L'empreinte carbone des serveurs, celle des ressources réseau et du système de refroidissement seront conjointement optimisées tout en intégrant le comportement dynamique des ventilateurs des équipements et les exigences des utilisateurs, en termes de requis de performance et de contraintes de co-localisation.

CHAPITRE 3 DÉMARCHES DE L'ENSEMBLE DU TRAVAIL DE RECHERCHE

Cette thèse vise à définir un cadre de placement de plusieurs applications à l'échelle d'un InterCloud, dans le but de réduire l'empreinte écologique d'un tel environnement informatique. Dans l'optique d'atteindre cet objectif, plusieurs étapes ont été prises en considération. Ainsi, dans ce chapitre, nous nous proposons de mettre en évidence les liens existant entre les objectifs de recherche, énoncés à la section 1.3, et les travaux présentés dans les chapitres subséquents. De manière générale, les travaux réalisés sont divisés en trois (3) grands volets et sont présentés dans ce document tel que soumis aux revues.

3.1 Volet 1 : Optimisation de l'empreinte carbone

Le premier objectif de la thèse consiste à développer un modèle de placement des applications dans l'optique de minimiser l'impact environnemental d'un InterCloud, en améliorant les modèles d'énergie existants et en intégrant au processus de placement, les contraintes de co-localisation et de performance des applications hébergées. Les deux objectifs suivants visent, pour leur part, à réaliser une analyse de la fonction objectif afin de déterminer l'importance de la détermination d'une valeur de température optimale, et à développer un algorithme de résolution exacte pour solutionner le problème de placement des applications et le comparer aux approches présentées dans la littérature. Ces trois (3) objectifs ont été atteints dans l'article intitulé « *On the Carbon Footprint Optimization in an InterCloud Environment* », présenté au chapitre 4, et dans lequel nous proposons un modèle de planification des VMs visant à minimiser l'empreinte carbone d'un environnement InterCloud.

3.1.1 Modélisation du problème

L'approche proposée permet de refléter le processus d'optimisation du placement des applications simples, à VM unique, dans l'optique de réduire l'impact environnemental d'un ensemble de data centers.

3.1.1.1 Puissance consommée et empreinte carbone

En effet, afin de faciliter son évaluation, l'empreinte carbone d'un data center a été traduite par la quantité de grammes de CO_2 rejeté dans l'atmosphère par celui-ci. Elle se calcule en combinant le facteur d'émission du data center, une constante qui dépend des sources

d'énergie utilisées pour alimenter le data center, et la consommation de puissance de ce dernier, modélisée comme la puissance des équipements informatiques actifs et la dissipation associée au système de refroidissement. Dans le cadre de ce premier volet, la charge à héberger a été représentée par des applications simples, chacune encapsulée dans une unique VM, et ne nécessitant aucune interaction entre elles. De ce fait, en termes d'équipements informatiques utilisés, seuls les nœuds de calcul ont été considérés. Ces derniers, de nature hétérogène, sont constitués d'un ensemble de châssis regroupant plusieurs serveurs-lames, de telle sorte que les serveurs d'un même châssis partagent les mêmes ventilateurs. Ainsi, la consommation totale d'un châssis est composée de sa puissance en mode veille, de la consommation des serveurs en activité et de la puissance des ventilateurs. Cette dernière découle de la loi des ventilateurs et varie au cube de la température d'entrée des châssis. Pour ce qui est de la puissance consommée par le système de refroidissement, un modèle basé sur la puissance des équipements informatiques et le coefficient de performance a été utilisé, lequel coefficient varie également en fonction de la température. Plus précisément, il a été observé qu'une augmentation de la température permet d'accroître l'efficacité du CRAC, ce qui se traduit par une diminution de la puissance consommée pour une même quantité de chaleur à faire évacuer. De ces comportements, découle une complexe interaction entre la consommation des châssis et celle du système de refroidissement en regard de la température fournie par ce dernier. Ceci nous porte alors à conclure que la consommation énergétique d'un data center, et l'empreinte carbone associée, dépend non seulement de la configuration résultant du placement des VMs, mais aussi de la température fournie par le CRAC.

3.1.1.2 Exigences des clients

De manière générale, un client donné peut imposer des restrictions, exigeant ainsi que certaines de ses VMs ne soient pas co-localisées sur le même serveur physique, particulièrement pour des raisons de redondance, ou encore qu'elles ne soient pas co-hébergées avec des applications de certains compétiteurs, par souci de sécurité et de confidentialité des données. Ces exigences sont alors recueillies, analysées et consignées dans des matrices d'interférence des VMs à placer, où les coefficients non nuls traduisent la présence d'une interférence entre deux VMs données. À des fins de simplification, nous supposons que le processus d'analyse entraînant la création des matrices d'interférence représente une étape antérieure, de telle sorte qu'elles constituent une entrée à notre modèle de placement. Ainsi, dans le cadre de notre travail, considérant ces matrices d'interférence, nous avons déduit des contraintes de co-localisation, sous la forme d'équations mathématiques.

3.1.1.3 Modèle de placement des VMs

Par la suite, dans le contexte d'une allocation statique, nous avons proposé un modèle mathématique global pour l'assignation des VMs aux serveurs, dans le but de réduire l'impact écologique d'un InterCloud. Ce modèle de placement, intra et inter data centers, considère les différents facteurs influençant l'empreinte carbone d'un tel environnement, et permet de résoudre le problème d'assignation tout en maintenant l'interaction entre ces différentes entités. En effet, la fonction objectif, de nature non linéaire, adapte le modèle d'empreinte carbone d'un unique data center, présenté précédemment, à un environnement de data centers géographiquement distribués et alimentés par différentes sources d'énergie vertes, semi et non renouvelables. Ainsi, afin de réduire l'impact écologique de cet environnement, il revient au modèle de déterminer la configuration idéale des VMs et les valeurs de température optimale à fournir par le système de refroidissement de chaque data center actif, de manière à ce que la consommation énergétique des data centers, combinée à leur facteur d'émission, résulte en un coût d'empreinte carbone minimal. Par ailleurs, ce processus de placement intègre également les contraintes de co-localisation imposées par les clients, tout en s'assurant d'une bonne isolation des performances des applications hébergées. En ce qui a trait aux performances des applications, une adaptation de l'approche présentée par Sharifi *et al.* (2012) a été développée afin de les modéliser, et dans le contexte de notre travail, le placement des VMs s'effectue sous la contrainte d'aucune dégradation de performance.

3.1.1.4 Étude analytique et reformulation du problème

Par ailleurs, les comportements antagonistes des ventilateurs des châssis et du système de refroidissement d'un data center donné, face à une hausse de température, et résultant, de surcroît, en une fonction objectif de nature non linéaire, nous ont amenés à analyser la pertinence de pouvoir déterminer, pour un data center donné, la température optimale à laquelle la puissance totale et l'empreinte carbone associée sont minimales. Ainsi, les résultats d'une étude de la première dérivée de la fonction objectif ont révélé que, dans l'intervalle de températures imposé par la ASHRAE, l'importance de la charge à héberger influence largement la monotonie de la fonction, en d'autres termes, dépendamment de la capacité totale des VMs à placer, la fonction objectif, dans l'intervalle de température d'intérêt, peut être strictement croissante, ou décroissante, ou même changer de signes. Dans ce dernier cas, une analyse de la seconde dérivée de la fonction a également été de mise afin de déterminer l'existence d'une possible valeur intermédiaire de température, entre 15° et 35° Celsius, comme température d'équilibre. Les résultats obtenus ont démontré que, pour une charge donnée et en l'absence de toute violation de performances, la fonction est toujours convexe dans l'intervalle d'inté-

rêt, ce qui suppose que, non seulement il existe une valeur de température permettant de minimiser la puissance et l'empreinte carbone d'un data center, mais aussi que cette valeur n'est pas nécessairement connue à priori, puisqu'elle peut se situer dans l'intervalle $[15° - 35°]$ Celsius. Ceci permet de mettre en évidence l'importance d'utiliser le modèle proposé, non seulement afin d'effectuer un placement des VMs, mais aussi dans le but de déterminer cette valeur optimale de température pour chaque data center actif, laquelle valeur dépend de l'infrastructure physique et de la charge hébergée.

Les modèles non linéaires étant souvent difficiles à résoudre, le problème d'optimisation de l'empreinte carbone d'un environnement InterCloud a été reformulé de manière à utiliser les solveurs linéaires et obtenir une solution exacte. La non-linéarité de la fonction objectif initiale étant liée à la température, en fixant la valeur de cette dernière pour chaque data center, le problème d'optimisation se trouve transformé en un modèle totalement linéaire. Ainsi, un algorithme de résolution exacte, *CB_OPT_SLA*, basé sur la programmation linéaire en nombres entiers, a été implémenté en AMPL où, pour chaque combinaison de valeurs de température des data centers, la version linéaire du modèle est résolu à l'aide du solveur CPLEX. Par la suite, la combinaison de températures pour laquelle l'empreinte carbone est minimale est retenue comme solution finale. En raison de certaines considérations pratiques, et suite à des expériences préliminaires, il nous a été possible de réduire l'intervalle de variation de la température, afin de limiter le temps d'exécution.

3.1.2 Évaluation de performance

Dans le but d'évaluer les performances du modèle d'optimisation proposé, plusieurs expériences ont été conduites. Pour mieux illustrer la prise en compte du comportement dynamique des ventilateurs des châssis, nous avons construit un modèle, *CB_OPT*, similaire au premier, mais faisant abstraction des contraintes d'interférence, de dégradation de performance et de température limite des châssis.

3.1.2.1 Structure du coût d'empreinte carbone dans un data center

Dans un premier temps, la structure du coût d'empreinte carbone a été analysée. Plus particulièrement, pour une charge fixe hébergée dans un data center, cette expérience a pu mettre en évidence les interactions complexes entre la consommation énergétique des châssis et celle du cooling, notamment en fonction de la température fournie par ce dernier. Le but étant d'évaluer la pertinence du modèle développé dans le processus de détermination de la température optimale d'équilibre. Les résultats ont pu démontrer que même à l'échelle d'un unique data center hébergeant une charge fixe, la valeur optimale de température n'est pas

triviale à déterminer, car l'empreinte carbone minimale s'obtient à une valeur de température bien précise, laquelle valeur dépend des caractéristiques physiques du data center en question et également de la charge hébergée. Ceci permet de confirmer la grande utilité du modèle d'optimisation proposé où le processus de placement des VMs et la détermination de la température optimale pour chaque data center doivent être réalisés de manière conjointe afin de réduire l'impact écologique d'un InterCloud.

3.1.2.2 Impact de la charge

Aussi, afin d'évaluer l'apport de la prise en compte du caractère hétérogène des VMs, nous avons également analysé l'impact de la charge sur l'évolution de l'empreinte carbone et la température optimale, notamment en comparant un premier scénario où, pour tous les jeux de données, toutes les VMs sont identiques entre elles, avec un second scénario, où les VMs sont totalement différentes les unes des autres. À des fins de simplification, cette expérience a également été conduite à l'échelle d'un unique data center où l'infrastructure physique a été gardée fixe. Cette expérience a pu démontrer qu'à l'inverse du premier scénario où l'empreinte carbone optimale augmente avec le nombre de VMs à placer, dans le cas d'une charge hétérogène, l'impact de la charge sur l'empreinte carbone et la température optimale est souvent imprévisible, ce qui vient encore souligner l'importance de l'approche proposée dans le but de déterminer la configuration idéale - placement des VMs et température optimale, afin de minimiser l'impact écologique d'un data center.

3.1.2.3 Performances du modèle à l'échelle d'un unique data center

CB_OPT a ensuite été comparé à trois modèles de référence utilisés dans la littérature et visant à minimiser la puissance ou l'empreinte carbone d'un unique data center : *CNS*, *CB_MIN_TEMP* et *CB_MAX_TEMP*. Le premier modèle réalise une consolidation des VMs et ignore tout du profil énergétique des équipements, le second modèle, *CB_MIN_TEMP* minimise l'empreinte carbone en considérant la borne inférieure de température, entraînant un PUE fixe d'environ 1.5, et le troisième modèle maximise l'efficacité du CRAC, avec une température de 35° Celsius. Comme attendu, *CB_OPT* génère de meilleures solutions en termes de coût d'empreinte carbone, car il réalise simultanément une consolidation efficace des VMs tout en déterminant la valeur optimale de la température, laquelle est souvent différente des valeurs triviales, 15° et 35° Celsius, utilisées respectivement dans *CB_MIN_TEMP* et *CB_MAX_TEMP*. Toutefois, bien qu'il permette d'économiser jusqu'à environ 30% en coût de carbone, en raison de la complexité du modèle proposé, *CB_OPT* demeure un problème difficile à résoudre, comme en témoignent les temps de calcul extrêmement élevés, et

le taux de solutions non évaluées, en raison des limites de ressources physiques, lequel taux peut aller jusqu'à 100%.

3.1.2.4 Impact de la puissance nominale des ventilateurs

Une analyse plus poussée de l'impact de la puissance des ventilateurs a été également effectuée. Les résultats ont pu démontrer que pour une charge et une infrastructure physique fixes, le coût optimal d'empreinte carbone augmente en fonction de la valeur de la puissance nominale des ventilateurs, alors que la valeur optimale de température diminue. Les simulations ont été poussées de manière à évaluer les gains réalisés, avec le modèle proposé, en fonction de la puissance nominale des ventilateurs. Suite aux résultats obtenus, il nous a été permis de conclure que *CB_MIN_TEMP* et *CB_MAX_TEMP* peuvent être utilisés pour réaliser l'optimisation de l'empreinte carbone respectivement pour des valeurs extrêmes de la puissance nominale, avec un taux d'erreur pouvant aller jusqu'à environ 8%. Toutefois, pour des valeurs intermédiaires de la puissance des ventilateurs, ces modèles ne peuvent plus convenir, comme le taux d'erreur augmente considérablement. Ces résultats permettent alors de démontrer l'importance de *CB_OPT* à pouvoir déterminer la configuration entraînant le coût de carbone minimal, peu importe la valeur de la puissance nominale des ventilateurs.

3.1.2.5 Performances du modèle à l'échelle d'un InterCloud

Afin de démontrer les performances de notre approche à l'échelle d'un InterCloud, l'algorithme de résolution exacte, *CB_OPT*, a également été comparé à deux autres modèles de référence qui s'appliquent aux environnements composés de plusieurs data centers. Le premier modèle, *INS*, consolide les VMs en considérant d'abord les data centers les plus verts, alors que la seconde méthode, *PWR*, tente de placer les VMs de manière à réduire le coût énergétique de l'InterCloud. Pour ces deux méthodes de référence, le placement a été réalisé suivant le critère d'optimisation d'intérêt, et l'empreinte carbone résultant de la configuration obtenue a été ensuite calculée. Les résultats ont démontré que le modèle *CB_OPT*, quoique plus long à résoudre, permet jusqu'à environ 65% d'économie en termes de coût d'empreinte carbone.

3.1.2.6 Comportement du modèle proposé face aux exigences des utilisateurs

Nous avons, par la suite, voulu mettre en évidence l'efficacité du modèle proposé à réaliser un placement des VMs, tout en respectant les exigences des utilisateurs en termes d'interférence ou de dégradation de performance, ou encore les contraintes liées aux performances des

équipements. Toutefois, en raison d'un espace limité, seuls les résultats en rapport avec les contraintes de co-localisation ont été illustrés dans cet article ; les autres résultats étant présentés au chapitre 7. Pour ce faire, nous avons comparé le modèle sans contraintes, *CB_OPT*, au modèle plus général, *CB_OPT_SLA*, considérant les contraintes d'interférence. Les deux modèles ont été, certes, comparés sur la base de la valeur d'empreinte carbone obtenue. Toutefois, pour mieux évaluer les mérites du modèle sans contraintes, *CB_OPT*, nous avons également considéré une métrique de performance, le coefficient de consolidation ou CF, mesurant l'aptitude d'un algorithme à déterminer une configuration optimale permettant de réduire la consommation de puissance tout en maintenant un niveau de performance adéquat. Comme attendu, le modèle sans contraintes, *CB_OPT*, génère des coûts d'empreinte carbone généralement moins élevés que le modèle incluant les contraintes d'interférence, *CB_OPT_SLA*. Toutefois, l'évaluation du coefficient de consolidation a permis d'indiquer que les solutions du modèle *CB_OPT* sont, en fait, beaucoup moins intéressantes, car le gain d'empreinte carbone résultant du processus de consolidation aveugle, ne permet pas de contrebalancer le pourcentage de violation des contraintes. Ceci résulte en des coefficients de consolidation extrêmement élevés, dont la valeur minimale est d'environ 18 comparée à une valeur idéale égale à l'unité. Ces résultats ont également permis de démontrer les bonnes performances du modèle global, *CB_OPT_SLA*, comparé au modèle sans contraintes, *CB_OPT*, dans le sens que le premier permet parfois d'obtenir des configurations de coût identique au modèle simplifié, en s'assurant, de surcroît, du respect des exigences des utilisateurs.

3.2 Volet 2 : Adaptation de l'heuristique de Recherche Locale Itérée ou ILS

Les résultats issus du premier volet ont aussi démontré que, même dans le cas du modèle simplifié, *CB_OPT*, les instances pouvant être résolues de manière exacte sont de petite taille. En raison de la nature NP-complet du problème, le temps de calcul croît de manière exponentielle en fonction des entrées. L'ajout de contraintes de co-localisation n'a fait que complexifier la résolution où, en dehors du temps d'exécution élevé, il a été remarqué qu'environ 40% des instances n'ont pas pu être résolues de manière optimale, en raison des limites physiques de la machine utilisée. Compte tenu de la complexité du problème, trouver des solutions optimales pour les grands jeux de données résultera en une explosion combinatoire, où le temps d'exécution risque d'être excessivement long, et la puissance de calcul, insuffisante. Une manière efficace de contourner le problème serait d'utiliser des méthodes approchées, lesquelles permettraient d'obtenir des solutions de qualité en un temps polynomial. Tels sont les sujets de nos objectifs 4 et 5, où il est question de proposer une méthode de résolution basée sur les métaheuristiques et d'en évaluer les performances, en termes d'excellent com-

promis entre temps de calcul et pertinence des solutions, afin de résoudre les problèmes de placement de VMs, particulièrement pour les instances de grande taille. Ces deux objectifs ont été atteints dans le chapitre 5, lequel présente l'article intitulé « *An Iterated Local Search Approach for Carbon Footprint Optimization in an InterCloud Environment* ».

3.2.1 Approche de résolution

La méthode de résolution proposée dans cet article, *ILS_CBF*, est basée sur l'heuristique de Recherche Locale Itérée ou ILS, qui part d'une solution initiale et, à chaque itération, parcourt le voisinage de la configuration actuelle afin d'en dégager une meilleure solution. Suite à l'arrêt prématuré de l'exploration, des perturbations permettent de repartir d'un nouveau point et de relancer la recherche. Dans notre adaptation du ILS, deux mécanismes de génération de la solution initiale ont été considérés : un processus d'assignation totalement aléatoire et une méthode de placement basée sur l'algorithme du FFD. Pour ce qui a trait au mécanisme de recherche locale, un processus de descente a été développé, dans lequel deux types de mouvements ont été considérés : le déplacement d'une VM d'un serveur à un autre, et l'assignation d'une nouvelle valeur de température à un data center actif. À chaque itération, l'algorithme de recherche locale accepte le meilleur mouvement qui diminue le coût de la solution actuelle. Afin d'accélérer le processus d'évaluation d'une configuration, des fonctions de gains, traduisant la différence entre le coût de la solution actuelle et celui du voisin considéré, ont été déterminées. Ces fonctions de gains incluent, non seulement, le gain associé au coût intrinsèque de la configuration, c'est-à-dire, le gain en termes d'empreinte carbone, mais également, les gains relatifs aux pénalités découlant de la violation des différentes contraintes associées au problème. Divers mécanismes de perturbation ont également été considérés afin d'éviter le piège des optima locaux : une méthode de placement aléatoire, un mécanisme de permutation et une approche visant, par exemple, à assigner les VMs aux serveurs inactifs. Ainsi, considérant ces différentes méthodes, trois modes de perturbation ont été développés : le premier fait intervenir une des trois méthodes de perturbation citées précédemment tout au long de l'exploration, le second mode sélectionne, de manière aléatoire, la méthode de perturbation à utiliser après chaque descente, et le dernier mode combine une méthode de perturbation avec un mécanisme de rappel.

3.2.2 Évaluation de performance

Les résultats présentés dans cet article découlent de deux types d'expérimentation : la paramétrisation et l'évaluation des performances de l'algorithme. Par souci de simplicité et sans perte de généralité, lors de l'implémentation de l'algorithme, les contraintes relatives à la

performance des applications et aux limites de fonctionnement des châssis ont été ignorées. De plus, nous avons considéré deux types de scénarios : le premier faisant abstraction des contraintes de co-localisation, et le second les intégrant. Toutefois, en raison d'un espace limité, seuls les résultats du premier scénario sont présentés dans cet article.

3.2.2.1 Paramétrisation

La première phase de simulation nous a permis de déterminer le mécanisme idéal à implémenter à chaque étape de l'algorithme de même que la valeur optimale des paramètres clés à utiliser pour l'heuristique *ILS_CBF*. Plus particulièrement, les résultats ont démontré que lorsque la solution initiale est générée à l'aide de l'algorithme FFD, le mécanisme de descente donne de meilleures solutions en temps CPU et coût d'empreinte carbone. Pour ce qui est des perturbations, les expériences ont été réalisées de manière progressive. Pour une valeur constante du pourcentage de perturbation et du nombre de relances, les différents mécanismes de perturbation (premier mode) ont été évalués. Les résultats ont démontré que la méthode de permutation permet d'obtenir le meilleur compromis entre le coût de la solution et le temps de calcul. Par la suite, considérant ce mécanisme, l'influence de l'intensité de la perturbation et du nombre de relances sur la qualité de la solution a également été évaluée. Ceci nous a permis de déterminer la valeur idéale pour chacun de ces paramètres clés. Enfin, partant de ces valeurs, les trois modes de perturbation ont été comparés entre eux : le premier mode considère la permutation comme perturbation, le second mode sélectionne l'un des trois mécanismes de perturbation, et le troisième mode implémente le processus de permutation à laquelle est combiné un mécanisme de diversification. Les résultats ont démontré que le troisième mode, soit la permutation intégrant la diversification, permet d'obtenir les meilleures solutions.

3.2.2.2 Méthode proposée et solution optimale

La solution obtenue après l'application complète de l'heuristique *ILS_CBF* est souvent un optimum local qui peut, dans certains cas, s'avérer être l'optimum global. Mais, à défaut de connaître d'avance l'optimum global, il est difficile de se prononcer sur la qualité de la solution obtenue. Dans ce contexte, une fois la phase de paramétrisation complétée, nous avons comparé les performances de l'algorithme *ILS_CBF* à celles obtenues avec la méthode exacte définie précédemment. Plus particulièrement, les résultats issus de l'implémentation de *CB_OPT* avec AMPL/CPLEX (voir le premier volet) ont été considérés lors de cette comparaison. Les résultats ont démontré que *ILS_CBF* trouve des solutions qui sont, en moyenne, à environ 0.2% de la solution optimale, avec un écart maximal de 2.6% par rapport

à la borne inférieure. De plus, suite à 10 exécutions successives de l'algorithme sur chaque taille de problème, *ILS_CBF* a pu obtenir la solution optimale pour près de 90% de ces jeux de données. Il nous a également été permis de noter que, dans le cas de l'heuristique *ILS_CBF*, le temps de calcul augmente linéairement en fonction de la taille du problème, contrairement à la croissance exponentielle observée pour la méthode exacte. Ainsi, avec un temps d'exécution moyen inférieur à 3 secondes, une valeur maximale avoisinant les 4 secondes et un coût d'empreinte carbone sensiblement égal à la valeur optimale, il appert que l'algorithme *ILS_CBF* proposé permet un excellent compromis entre la qualité de la solution et le temps d'exécution.

3.2.2.3 Méthode proposée et autres algorithmes de référence

Afin d'analyser les performances générales de *ILS_CBF* et de mettre en évidence son efficacité, 10 exécutions de l'algorithme ont été réalisées sur des instances de différentes tailles, incluant celles ne pouvant être résolues optimalement avec le solveur CPLEX. Les résultats obtenus ont été comparés à ceux découlant de l'implémentation de trois méthodes approchées retrouvées dans la littérature : la première, Green_Data_Center_First$_{Improved}$ ($GDCF_I$), qui utilise une version modifiée du BF afin de placer les VMs d'abord dans les data centers les plus écologiques ; une version adaptée de l'algorithme FFD (FFD_A), et une modification du BFD (BFD_A). Les résultats ont pu démontrer que l'heuristique proposée, *ILS_CBF*, donne toujours de meilleures configurations, en termes de coût d'empreinte carbone. En effet, le caractère glouton des algorithmes de comparaison leur permet, certes, de déterminer rapidement un schéma de configuration, toutefois il n'existe aucune garantie quant à la qualité des solutions obtenues avec ces méthodes, notamment parce qu'aucun mécanisme d'exploration de l'espace de recherche n'est implémenté par ces approches. Ainsi, avec une économie de coût de carbone pouvant s'élever jusqu'à 34%, et un temps de convergence polynomial, les résultats ont pu mettre en évidence les bonnes performances de l'heuristique proposée, en termes de compromis entre qualité de la solution et temps d'exécution.

3.3 Volet 3 : Modèle global d'empreinte carbone et adaptation d'une méthode de résolution hybride

Dans l'optique d'une réduction de l'empreinte environnementale d'un InterCloud, les deux premiers volets de la thèse s'étaient attardés sur le placement de VMs fonctionnant, chacune de manière indépendante, les unes des autres. Toutefois, des applications complexes, s'étendant sur plusieurs VMs, se font de plus en plus héberger dans le Cloud. Avec le trafic inter-VMs introduit par ces applications, les ressources réseau, particulièrement, les com-

mutateurs et les liens physiques, sont souvent sollicitées afin de faire transiter l'information d'une VM à une autre. Or, comme la consommation énergétique des ressources réseau représente environ le quart de la puissance totale d'un data center, il en résulte alors que l'impact énergétique de ces équipements ne saurait être négligé plus longtemps, lorsque vient le temps de décider de l'emplacement des VMs afin de réduire l'empreinte écologique de plusieurs data centers. En ce sens, l'avant-dernier objectif de la thèse vise à étendre le modèle de placement des VMs présenté dans les deux premiers volets, afin de considérer un modèle d'assignation plus complet, où des applications complexes, notamment leurs VMs et trafics, de même que les contraintes s'y rattachant, sont considérées. De plus, le processus de routage du trafic étant également NP-complet, la combinaison de ce dernier au mécanisme de placement des VMs résulte en un problème encore plus difficile. L'utilisation d'une méthode approchée est donc encore de mise afin de solutionner les instances de grande taille. Tel est le sujet du dernier objectif de la thèse, où il est question de proposer et d'évaluer une méthode de résolution à base de métaheuristiques afin de résoudre les problèmes avec le modèle global d'empreinte carbone proposé. Ces deux derniers objectifs ont été atteints dans le chapitre 6 qui présente l'article intitulé « *A Hybrid Approach for Optimizing Carbon Footprint in InterCloud* ».

3.3.1 Démarche

Dans ce dernier volet, la démarche est divisée en 2 étapes : la première phase a pour but de modéliser le processus de placement des applications simples et complexes, alors que la seconde phase propose une approche visant à résoudre le problème de minimisation de l'empreinte carbone, particulièrement pour les instances de grande taille.

3.3.1.1 Intégration du trafic inter-VMs

Le modèle proposé dans ce volet, $CB_G\_OPT\_SLA$, représente entièrement le problème d'optimisation de l'empreinte carbone dans un environnement InterCloud. En effet, le modèle présenté précédemment a été repris et amélioré afin de considérer les deux types d'applications : des VMs uniques ou encore des applications regroupant un ensemble de VMs et les demandes en trafic qui s'y rattachent. En plus des contraintes de co-localisation associées aux VMs, d'autres requis, en termes de data centers potentiels pour certaines applications complexes, ont également été introduits dans ce modèle. Cette approche permet d'illustrer le fait que, par souci de sécurité ou pour des questions de territorialité, des applications sensibles ne peuvent être hébergées dans certaines régions bien définies. Aussi, il a été retenu que toutes les composantes d'une même application doivent être hébergées dans le même data center, ceci afin d'éviter d'introduire des délais d'interactions entre les différentes entités d'une même

application localisées à des endroits différents - le délai à l'intérieur d'un data center étant négligeable. Pour ce qui est du schéma d'interconnexion des serveurs d'un data center, la topologie en arbre a été utilisée en considérant l'existence d'un unique chemin entre chaque paire de machines physiques. Ainsi, la fonction objectif doit alors minimiser l'empreinte carbone totale incluant l'impact des châssis et serveurs, celui du cooling et des équipements réseau. L'optimisation de ces trois entités étant souvent de nature contradictoire, ce problème est souvent abordé sous le biais d'une optimisation multi-objective. Toutefois, nous avons opté pour une agrégation des différents objectifs avec des poids identiques pour chacun d'eux.

3.3.1.2 Approche de résolution

Un algorithme basé sur les métaheuristiques a également été proposé. Cet algorithme, dénommé ITS_G_{CBF}, résulte de l'hybridation de deux heuristiques connues, le Iterated Local Search et le Tabu Search. Plus particulièrement, ITS_G_{CBF} implémente l'algorithme ILS où le mécanisme de recherche locale est une adaptation de l'heuristique TS. Cette hybridation permet de tirer profit des avantages des deux méthodes : TS permet d'explorer efficacement l'espace de recherche à l'aide des mécanismes de mémoire permettant d'éviter les cycles et les pièges des optima locaux, alors que ILS est utilisé pour guider la recherche vers de nouveaux voisinages à l'aide d'opérateurs de perturbation qui relancent l'exploration à partir d'une nouvelle configuration de départ. L'algorithme proposé présente deux phases de recherche locale, combinées à deux niveaux de perturbation. ITS_G_{CBF} débute avec une solution initiale à laquelle un premier opérateur de recherche locale, TS_1, est appliqué. Ce dernier définit, par exemple, des mouvements de déplacements d'une VM d'un serveur à un autre serveur du même data center. Un second mouvement permet également de changer la température d'un data center d'une valeur à une autre. Afin de faciliter l'évaluation d'une configuration, deux fonctions de gains ont été considérées où, pour chaque solution, des gains relatifs aux violations de contraintes sont combinés au gain associé au coût d'empreinte carbone. Afin d'enrayer les cycles et d'éviter les minima locaux, deux listes taboues ont également été implémentées. Lorsque cette recherche stagne, une nouvelle méthode de recherche locale, TS_2, est activée. Cette dernière, implémente des mécanismes sensiblement identiques à TS_1, mais fait intervenir des mouvements de permutation de VMs à l'échelle d'un même data center, ou encore en échangeant les températures de deux data centers différents. Il arrive que la recherche n'avance plus, car l'exploration est effectuée au sein d'un unique data center. Aussi, afin de relancer la recherche, les perturbations sont déclenchées. La première phase consiste à modifier la meilleure solution et à relancer l'exploration à l'aide de TS_1 pour un certain nombre d'itérations. Lorsque la recherche stagne à ce niveau, la meilleure solution est alors

perturbée, mais cette fois, suivie d'une exécution de l'algorithme TS_2. Cette seconde phase de perturbation est invoquée tant et aussi longtemps qu'il est permis d'améliorer la solution courante. Généralement, l'opérateur de perturbation détermine un certain pourcentage d'applications et de VMs à modifier. Puis, pour chaque VM simple, un nouveau data center, différent de l'hôte actuel, est choisi et la VM est placée sur un serveur de manière aléatoire. Pour ce qui est d'une application, un nouveau data center est sélectionné dans la liste des data centers potentiels de cette application. Une fois le data center choisi, chaque VM de l'application est aléatoirement assignée à un serveur et par la suite, le trafic est routé. Il peut également arriver qu'en dépit des différents mécanismes implémentés, on observe une certaine stagnation. Afin de relancer l'exploration dans des voisinages non explorés, un mécanisme de diversification a également été introduit.

3.3.2 Évaluation de performance

Dans la phase d'expérimentation, autant le modèle global que la méthode de résolution proposés ont été évalués.

3.3.2.1 Modèle global d'empreinte carbone et autres modèles de référence

Un algorithme a été implémenté en AMPL/CPLEX en utilisant les techniques de programmation linéaire en nombres entiers afin de résoudre, de manière exacte, le modèle global proposé et comparer ses performances à celles d'autres modèles de référence. À ce stade, afin de faciliter les comparaisons, le modèle global a été simplifié en faisant abstraction de certaines contraintes. Afin de mettre en évidence les avantages d'une optimisation conjointe, le modèle proposé a été comparé à trois autres approches : un modèle qui optimise seulement le placement des VMs (VM_{OPT}), un second qui s'attaque au routage du trafic (NET_{OPT}), et un troisième modèle qui tente de maximiser l'efficacité énergétique (EFF_{OPT}). Comme attendu, le modèle proposé permet d'obtenir de meilleures configurations en termes de coût d'empreinte carbone, avec un gain pouvant s'élever jusqu'à environ 900% pour les instances considérées. Cependant, il est à noter que cette optimisation se fait au prix d'un temps de calcul, en moyenne, relativement plus élevé, en raison de la complexité du modèle proposé. Toutefois, l'économie en carbone réalisée étant substantiellement plus importante comparée aux écarts en temps de calcul observés, les résultats ont pu démontrer la grande efficacité du modèle proposé par rapport aux modèles réalisant une optimisation mono-objectif.

3.3.2.2 Paramétrisation

La méthode de résolution approchée proposée, ITS_G_{CBF} a été implémentée en C++ et son efficacité en termes de compromis entre coût d'empreinte carbone et temps d'exécution a été évaluée à travers plusieurs simulations. Une fois de plus, par souci de simplicité et sans perte de généralité, seules les contraintes de co-hébergement des VMs et celles liées à l'emplacement limité des applications ont été considérées lors de l'implémentation. Aussi, deux scénarios ont été considérés : le premier, sans contraintes de co-localisation, et le second, intégrant ces dernières. Les résultats du second scénario n'ont pas été présentés dans cet article, mais certains sont illustrés au chapitre 7. Des expériences préliminaires nous ont permis de dégager les valeurs optimales des paramètres clés de la méthode, dont la plupart sont liées à la taille du problème. Afin de mettre en évidence les bonnes performances de ITS_G_{CBF}, ce dernier a d'abord été comparé à une borne inférieure, pour de petites instances, et ensuite, à des méthodes de résolution approchées, pour les problèmes de plus grande taille.

3.3.2.3 Méthode proposée et solution optimale

Nous avons comparé les solutions obtenues par l'algorithme ITS_G_{CBF} avec une borne inférieure, OPT_G_{CBF}, calculée en résolvant le problème de placement global à l'aide de la méthode exacte en utilisant le solveur CPLEX. Les résultats montrent, qu'en termes de coût d'empreinte carbone, l'algorithme proposé est en mesure de trouver des solutions qui sont, pour la plupart, à moins de 3% de la borne inférieure, et où la solution optimale a pu être trouvée pour 35% des instances. Une autre comparaison a également été effectuée en regard du temps d'exécution des deux méthodes. En général, les résultats démontrent que le temps de calcul de ITS_G_{CBF} présente un profil linéaire par rapport à la taille du problème, alors que la méthode exacte affiche une croissance exponentielle. Plus particulièrement, alors que pour ITS_G_{CBF}, les temps d'exécution sont inférieurs à 110 secondes, OPT_G_{CBF} peut aller jusqu'à 30 heures pour résoudre le problème le plus difficile. Avec un écart moyen par rapport à la borne inférieure avoisinant les 3% et un temps d'exécution moyen inférieur à 105 secondes, les résultats ont su clairement souligner les bonnes performances de l'algorithme ITS_G_{CBF}, par rapport à la méthode exacte, OPT_G_{CBF}, en termes de compromis entre la vitesse d'exécution et le niveau de convergence vers des solutions de bonne qualité.

3.3.2.4 Méthode proposée et autres algorithmes de référence

Afin d'évaluer le comportement général de ITS_G_{CBF}, nous avons évalué les performances de l'algorithme en l'exécutant sur différentes tailles de problèmes, pouvant être résolues op-

timalement par CPLEX ou pas. Les résultats obtenus ont été comparés à ceux découlant de l'implémentation de quatre (4) autres algorithmes de référence. Le premier algorithme, *RAND*, introduit un mécanisme d'assignation aléatoire, et les trois méthodes sont des variantes de la méthode ITS_G_{CBF} proposée. La première variante, *ILS*, est ITS_G_{CBF} avec un mécanisme de descente et sans diversification ; la seconde, *TS*, implémente uniquement le premier mécanisme de recherche locale, TS_1 ; et la dernière variante, *ITS_RAND*, reprend l'implémentation du ITS_G_{CBF}, mais où la solution initiale est générée de manière aléatoire. Les solutions obtenues ont pu démontrer que la méthode proposée ITS_G_{CBF} est toujours en mesure de trouver les configurations de coût minimal. La différence est particulièrement notable dans le cas de l'algorithme *RAND* où la configuration est déterminée de manière aléatoire, sans exploration plus poussée de l'espace de recherche. Ceci résulte, souvent en des configurations où l'empreinte carbone est excessivement élevée, ou encore en des solutions infaisables, en raison de la violation de certaines contraintes de capacité. Bien que l'écart entre ITS_G_{CBF} et ses deux premières variantes, *ILS* et *TS*, soit faible, l'hybridation de ces deux méthodes permet toujours d'améliorer les coûts de solution en termes d'empreinte carbone. Toutefois, il était possible d'observer qu'en dépit du fait que la dernière méthode de référence tire profit de ces deux types d'opérateurs, elle ne permet pas toujours d'obtenir de bonnes solutions. En effet, avec une configuration initiale déterminée de manière aléatoire, il n'existe aucune garantie quant à la qualité et à la faisabilité de cette solution initiale, ce qui rend l'exploration de l'espace de recherche très difficile. Une comparaison plus poussée entre cette dernière variante et la méthode proposée a démontré que *ITS_RAND* présente un caractère moins évolutif que ITS_G_{CBF}, où en plus d'un coût d'empreinte généralement plus élevé, on observe que l'écart, en pourcentage, augmente avec la taille du problème. Nous avons également étudié le niveau de convergence des deux approches, en termes de vitesse d'exécution et qualité de la solution obtenue. Les résultats issus de cette analyse ont démontré que la méthode proposée produit de meilleures performances, non seulement en termes de coût de carbone, mais également en durée d'exécution. L'ensemble de ces expériences ont démontré que ITS_G_{CBF} est l'outil de résolution à utiliser afin de solutionner le problème global de placement des applications. En effet, la méthode fait montre d'une très grande flexibilité lui permettant d'explorer efficacement l'espace de recherche afin d'obtenir de bonnes solutions en un temps raisonnable.

3.4 Conclusion

En somme, les objectifs énoncés à la section 1.3 ont été atteints dans les quatre chapitres suivants. De manière générale, l'ensemble de notre travail permettra à un fournisseur d'infra-

structure physique, de mieux cerner le problème de placement des applications, autant simples que complexes, à l'échelle de son environnement InterCloud, en considérant conjointement l'impact écologique des nœuds de calcul, celui des ressources réseau et également l'empreinte du cooling. En plus des retombées sur le plan scientifique, cette thèse a également des répercussions sur le plan social et environnemental, en permettant aux propriétaires de data centers de réduire leur empreinte écologique tout en respectant les exigences des utilisateurs en termes de contraintes de performances, et en matière de sécurité et de redondance des données hébergées.

CHAPITRE 4 ARTICLE 1 : ON THE CARBON FOOTPRINT OPTIMIZATION IN AN INTERCLOUD ENVIRONMENT

Auteurs : Valérie Danielle Justafort, Ronald Beaubrun et Samuel Pierre.
Revue : Accepté pour publication dans le journal *IEEE Transactions on Cloud Computing*, en Octobre 2015.

Abstract

In this paper, we address the problem of Virtual Machine (VM) placement in an InterCloud with regard to the reduction of carbon footprint in such computing environment. In order to minimize data centers' Greenhouse Gas (GhG) emissions, this paper proposes a mathematical formulation, where the placement approach is stated as a Mixed-Integer Nonlinear Programming (MINLP) problem which aims at minimizing the carbon footprint of the InterCloud. The proposed formulation presents an accurate carbon footprint evaluation based on joint optimization techniques, such as smart and performance-aware workload consolidation, along with cooling efficiency maximization, while considering the greenness factor of the data centers and the dynamic behavior of the equipment cooling fans. Simulations with an exact method showed that, compared with other baseline approaches, the proposed mathematical model leads to optimal configurations with minimal GhG emissions in a single cloud, as well as in an InterCloud environment, and can yield savings of up to 65%. The proposed model has also been compared with an approach that performs blind VM consolidation, and the obtained results showed that such model can simultaneously lead to VM configuration that yields the minimum carbon footprint while performing smart VM consolidation in order to prevent Service Level Agreement (SLA) violations.

4.1 Introduction

Nowadays, Cloud Computing provides innovative computing solutions (Zhu et Ammar, 2006), (Chowdhury *et al.*, 2009), allowing small businesses to rent virtual servers as a service. However, its adoption comes with challenges, such as the Virtual Machine (VM) placement problem (Cardosa *et al.*, 2009), (Hermenier *et al.*, 2009), which tends to find the most convenient hosts for the Virtual Machines (VMs), based on optimization criteria. Until recently, the VM placement problem has been focusing on improving metrics, like performance and service availability (Dupont *et al.*, 2012). However, energy efficiency has lately become a great

concern (Mazzucco *et al.*, 2010), as data centers' power consumption has exponentially grown due to the intensive usage of large scale applications (Moore *et al.*, 2005), (Greenpeace, 2010). With the acknowledgment that Cloud carbon footprint is about 2% of Greenhouse Gas (GhG) emissions (Moghaddam *et al.*, 2012), and is constantly growing, the reduction of Information and Communication Technology (ICT) sector environmental impact has started to attract a great deal of attention .

In an InterCloud, composed of geographically distributed data centers (Moghaddam *et al.*, 2012), (Moghaddam *et al.*, 2011), the power consumption does not always reflect the carbon footprint: clean data centers' carbon footprint may be low, whereas their power consumption is very high. However, for semi-clean or non-clean data centers, a greenness factor has been introduced in Moghaddam *et al.* (2011) for computing the equivalent carbon footprint of a data center, based on its power consumption. Therefore, in an InterCloud composed of data centers powered by a mixture of energy sources, carbon footprint minimization may only be performed by jointly considering data centers' greenness factor and the power consumption issue.

Moreover, multiple management methods for computing nodes (Von Laszewski *et al.*, 2009), and workload consolidation techniques have been largely implemented by cloud providers in order to address the issue of high power consumption of data centers and their associated carbon footprint. As cooling power consumption accounts for about 30% of the total power consumed (Pakbaznia et Pedram, 2009), various works have also focused on maximizing the data center efficiency by increasing the data center indoor temperature, as suggested by a study from American Society of Heating, Refrigerating and Air Conditioning Engineers (ASHRAE) (ASHRAE, 2011). However, recent experiments have revealed that server fans will typically respond to a demand of increased airflow as inlet temperature rises, resulting in an increase of server power consumption (Moss et Bean, 2009). This dynamic behavior of IT equipment fans can negate the cooling gains associated with high temperatures. Therefore, task assignment that ignores such temperature dependancy will not necessarily result in minimizing both power consumption and carbon footprint.

Although VM consolidation helps tackling the problem of resource wastage, it often causes resource contention, leading to longer execution time and performance degradation (Sharifi *et al.*, 2012). In this context, recent works have proposed workload-aware consolidation techniques that ensured performance isolation. However, co-location constraints regarding security or redundancy, may arise, and should also be considered in the VM consolidation process (Bonde, 2010).

The challenges related to the carbon footprint optimization in an InterCloud lead to a number

of questions. Which data centers should be used considering their greenness factor? How to efficiently maximize resource utilization and perform workload consolidation without violating Quality of Service (QoS) requirements? And how to select the optimal temperature inside each active data center in order to balance the power consumption of the cooling system and the IT equipment?

In order to address the challenges stated above, this paper proposes an InterCloud planning model, named *CB_OPT_SLA*, that optimizes the VM placement and increases the data center efficiency with the purpose of reducing the total carbon footprint of such computing environment. Also, an InterCloud has been considered, instead of a single data center setup, in order to provide a more general model that can be applied regardless of the use case analyzed. Therefore, even though our work targeted an InterCloud environment, it can be easily adapted to a single cloud, powered by a clean energy source or not. The main contributions of this paper are listed below:

— We address the carbon footprint issue in an InterCloud environment and improve existing models by taking into account the heterogeneous nature and the dynamic behavior of the IT equipment. We also perform smart workload consolidation by considering VM performance degradation, as well as other consolidation constraints related to security or redundancy;

— Using modeling techniques for Integer Programming (IP) models (Williams, 2013), the VM placement problem has been stated as a Mixed-Integer Nonlinear Programming (MINLP) problem in order to minimize the carbon footprint in an InterCloud environment;

— An analysis of the monotony and the convexity of the problem has been done in order to prove that it is of a central importance to determine each data center's optimal temperature, using the proposed model, as the latter temperature is not trivial to be found although considering a bounded search space for the temperature interval;

— An Integer Linear Programming (ILP) formulation of the problem has been proposed, in order to use ILP solver packages for computing the optimum carbon footprint value.

The rest of the paper is organized as follows. Section 4.2 discusses relevant works related to the problem. The system model is presented in Section 4.3, whereas the problem formulation is described in Section 4.4. Section 4.5 presents the convexity analysis of the objective function followed by the problem re-statement. Experiments and numerical results are illustrated in Section 4.6, whereas conclusion and future works are presented in Section 4.7.

4.2 Related work

The power consumption of a distributed cloud environment does not always reflect its carbon footprint (Moghaddam *et al.*, 2012): green data centers will always have zero emission, regardless of their energy consumption, whereas in the case of data centers powered by semi-clean or brown energy sources, their carbon footprint becomes highly correlated with their power consumption (Moghaddam *et al.*, 2011) due to their non negligible greenness factor (Lenzen, 2010). Therefore, in the following, we will briefly discuss the relevant prior works regarding the energy consumption as well as the carbon footprint optimization problem.

Techniques, such as Dynamic Voltage Frequency Scaling (Burd *et al.*, 2000), (Miyoshi *et al.*, 2002) and CPU idle power saving states (Barroso et Hölzle, 2007), were developed to increase servers' energy efficiency (Liu *et al.*, 2009), (Dharwar *et al.*, 2012), (John, 2014). For instance, in (Li, 2015), Li proposed to dynamically adjust the server core power and speed in order to reduce energy consumption. However, this work only focused on hardware power management. Other works, such as (Cardosa *et al.*, 2009), (Li *et al.*, 2009), (Buyya *et al.*, 2010), (Feller *et al.*, 2011) and (Dong *et al.*, 2013), provided consolidation tools to reduce the amount of in-use hardware. In their approach, Liu *et al.* (2015) considered a fully parallelizable workload as well as identical servers, unlike CB_OPT_SLA, and proposed different algorithms to geographically balance the load and reduce both delay and energy cost. However, in large data centers, this homogeneity assumption does not hold, and minimizing the number of active equipment becomes unsuitable. Therefore, Kansal *et al.* (2010) introduced a model for VM power metering, based on which, approaches for placing VMs with regard to the energy minimization were developed (Buyya *et al.*, 2009), (Dupont *et al.*, 2012), (Larumbe et Sanso, 2013) and (Shuja *et al.*, 2014).

As the cooling cost accounts for about 30% of a data center's power cost (Rasmussen, 2007), heat extracting methods (Patel *et al.*, 2003), (Sharma *et al.*, 2005) were proposed in order to reduce the cooling energy. Moore *et al.* (2005) explored a temperature-aware workload placement method in order to minimize heat recirculation. Tang *et al.* (2006) proposed a heat flow model which has been used for optimizing the energy cost under server inlet temperature constraints (Polverini *et al.*, 2014), or for minimizing both server and cooling power of a data center (Pakbaznia et Pedram, 2009). Recent studies (Moss et Bean, 2009) revealed that server fans typically respond to a demand of increased airflow, as inlet temperature rises, resulting in an increase of the server power consumption. Based on these findings, Lee (2012) presented a case study of the impact of data center inlet temperature on energy efficiency, but no VM placement process was taken into account.

Moreover, some researches focused on the carbon footprint reduction problem. Liu *et al.* (2012) proposed a workload placement approach based on the available renewable energy sources and the cooling efficiency in order to minimize the operational cost of a data center. Goiri *et al.* (2015) maximized the use of green energy while meeting jobs' deadline by considering a mechanism that predicts the availability of solar energy. However, unlike *CB_OPT_SLA*, the carbon footprint reduction problem is tackled considering a single-data center setup and beside determining the best mix of renewable/brown energy sources, no other optimization mechanism was considered in order to increase a data center efficiency whenever green energy is unavailable.

Other works also considered the problem of minimizing the environmental impact of an InterCloud with data centers powered by different types of energy sources (Moghaddam *et al.*, 2011), (Moghaddam *et al.*, 2012) and (Van Heddeghem *et al.*, 2012). Although a carbon footprint model, based on the binomial distribution was introduced in (Van Heddeghem *et al.*, 2012), it cannot accurately evaluate the ecological impact of an InterCloud. Liu *et al.* (2015) also explored the benefits of geographical load balancing with regards to carbon emissions. However, carbon footprint was not explicitly considered as they put a dollar cost on the carbon emissions in their analysis. The authors also proposed an algorithm to find the optimal percentage of wind/solar energies in order to reduce the brown energy consumption in (Liu *et al.*, 2011). Moghaddam *et al.* (2012) and Moghaddam *et al.* (2011) also proposed a mathematical model for optimizing the total carbon footprint in a distributed cloud. Doyle *et al.* (2013) developed a method based on Voronoi partitions that schedules the workload in order to simultaneously reduce the delay, the energy and the carbon footprint cost. Khosravi *et al.* (2013) addressed the problem of increase in carbon footprint of a computing environment while considering the greenness factors of the cloud sites, their efficiency or Power Usage Effectiveness (PUE) and the servers' power profiles. Although numerous efforts were provided to reduce the brown energy consumption, these works differ from *CB_OPT_SLA* as only the workload placement was performed, with no consideration regarding the optimization of the cooling cost nor the dynamic behavior of the servers' cooling fans.

Also, due to their utilization patterns, co-allocated workloads often fight for the use of shared resources (Sharifi *et al.*, 2012), which results in performance degradation. In this context, Kusic *et al.* (2009) tried to minimize both power consumption and Service Level Agreement (SLA) violations in order to maximize the provider's profit. In (Srikantaiah *et al.*, 2008), an algorithm that performs VM placement based on energy-performance optimal points were proposed. Sharifi *et al.* (2012) introduced a Consolidation Fitness (CF) metric defined by the ratio of the SLA violation percentage due to consolidation to the amount of saved energy percentage gained through consolidation, in order to evaluate the merit of

a consolidation. In (Kim *et al.*, 2013), the VM placement algorithm used an interference model that determines the mutual impact of co-located applications. In general, most of the aforesaid researches focused on the workload characterization in order to place the VMs. However, the consolidation process cannot only be guided by the intrinsic performance of the co-allocated VMs, as other interference types related to security, data privacy or redundancy can alter the QoS expressed in the SLA.

Due to the heterogeneous nature of the data centers, the VM placement is no longer straightforward, as only the most power-efficient hosts should be used. Furthermore, smart VM consolidation should be performed in order to avoid any type of interference or performance degradation. Moreover, in order to increase data center efficiency, it becomes important to explore the power reduction of the Computer Room Air Conditioner (CRAC) against the potential increase in IT power, considering that for temperature-dependent equipment, energy savings can only be achieved within a temperature sweet spot. Although the previous approaches tried to reduce either the total power consumed or the environmental impact of the ICT sector, they lack an accurate mathematical formulation of the optimization problem, that should simultaneously consider the workload consolidation along with the interference constraints, the cooling power dissipation, the dynamic behavior of the IT equipment and the cleanness of the data centers.

Based on the above considerations, although the instinctive approach of packing as many VMs as possible in the cleanest data centers first seems straightforward, no guarantee can be made regarding the optimal solution of the whole process. By considering all the important parameters, we eliminate this short-sighted aspect of the VM planning. To the best of our knowledge, no prior work was able to tackle these previously mentioned issues altogether and formulate the problem using IP techniques in order to find the optimal solution. Therefore, we propose a mathematical formulation of the VM placement problem that will integrate multiple aspects, with the purpose of providing a global model. Even if the latter is more complex to solve, it has the merit of considering all the interactions and offers a complete solution by solving the VM placement problem as a whole.

4.3 System model

Several models are needed to illustrate our work. First, the InterCloud environment and a data center configuration are presented. Second, the workload representation and the associated SLA constraint models are described.

4.3.1 InterCloud environment

The InterCloud, as illustrated in Fig. 4.1, is a computing environment made of three components: the Cloud provider using a VM Management System to supply a pool of resources to clients, a set of data centers hosting the VMs, and the clients with their applications. A data center is composed of rows of racks with several chassis, each of which comprises multiple servers that share the same fans and power supply. The servers are heterogeneous, provide physical resources in terms of CPU cycles, memory size and disk, and can run multiple VMs. Furthermore, in this paper, we consider the chassis level as the temperature spacial granularity, as in (Pakbaznia et Pedram, 2009), and due to physical separators (Posladek, 2008), the chassis inlet temperature only depends on the cold air supplied by the CRAC system.

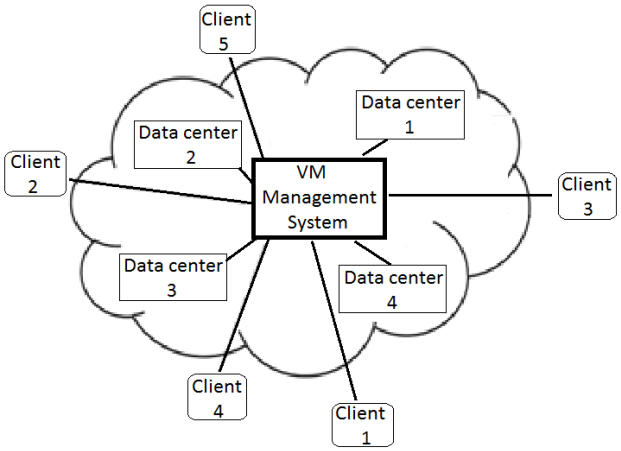


Figure 4.1 An InterCloud environment

4.3.2 Client and application representation

Cloud customers submit their applications, each of which is encapsulated in a unique VM, uploaded into the InterCloud system and deployed among the servers. The VM physical requirements are also characterized by a multi-dimensional resource vector consisting of the following components: [*cpu*, *disk*, *memory*]. In this paper, applications are running in a separated manner, therefore, no inter-VMs traffic will be considered.

4.3.3 SLA's constraints

In the VM consolidation process, several virtualization requirements, related to VM interference or intrinsic performance degradation, may arise.

4.3.3.1 Interference requirements

Interferences between VMs require that the latter should not be co-located. Interferences between VMs of the same client may be due to technical constraints (Bonde, 2010), for redundancy purpose or for ensuring service continuity. Therefore, in order to avoid blind VM consolidation, the clients submit their requirements in terms of VM-VM co-location constraints, based on which, matrices as in Table 4.1, are derived. In this table, a negative coefficient indicates a negative interference, while a null coefficient supposes that these VMs can be co-located or not. Also, interferences between VMs of different clients may occur if, for business purposes, such as privacy or security matters, a client imposes to keep his VMs away from competitors' VMs. Therefore, a VM-Client interference matrix is also computed, based on the clients requirements.

4.3.3.2 Performance requirements

In order to ensure proper execution of consolidated workloads, the approach regarding performance degradation presented in (Sharifi *et al.*, 2012) has been used, based on which, two conclusions can be stated: first, performance degradation due to CPU intensive workload consolidation is negligible, while disk intensive workloads are not meant to be consolidated; second, in the case of mixed workload consolidation, performance degradation occurs when the host server CPU utilization exceeds a certain value. From these conclusions, and considering that no degradation of performance is allowed, two main VM placement constraints have been defined: (1) a server can host up to one disk-intensive application; (2) for mixed workloads, the CPU utilization of a server should not exceed a threshold value.

4.4 Problem statement

In this section, an overview of the VM placement process is presented, followed by a formal definition of the mathematical model related to the optimization problem.

Table 4.1 Example of a VM-VM interference matrix of a client

IDs	VM_1	VM_2	VM_3	VM_4
VM_1	0	0	-1	-1
VM_2	0	0	0	-1
VM_3	-1	0	0	0
VM_4	-1	-1	0	0

4.4.1 VM placement overview

As the carbon footprint of a non-green data center is highly related to its power consumption (Moghaddam *et al.*, 2011), carbon footprint minimization for a data center does involve power consumption reduction, which is not as straightforward as it seems. In fact, due to the high consumption of the chassis base power (Pakbaznia et Pedram, 2009), and the heterogeneity of the servers, VMs should be assigned to the most power efficient equipment, so the remaining ones can be switched off. Also, smart consolidation should be performed in a way to prevent SLA violations. Furthermore, in order to improve data center efficiency and to reduce the cooling power consumption, the ASHRAE committee has recommended to increase the data center supplied temperature. However, the dynamic nature of IT equipment cooling fans may negate the cooling system gains. Therefore, selecting the optimal temperature to balance the fans' power consumption and the cooling power dissipation is highly desirable. The carbon footprint minimization can be therefore realized by jointly 1) considering the greenness factor of the data centers; 2) performing chassis and server consolidation, while respecting the SLA requirements; 3) determining the optimal value of the supplied temperature inside each active data center. Also, using IP techniques, the VM placement process will be stated as a MINLP problem.

4.4.2 Assumptions

We make the following assumptions in the VM placement process:

— The data centers have different greenness factors;
— All servers and chassis are initially off;
— The servers and VMs are heterogeneous;
— The data centers' features are known;
— The clients requirements in terms of number and types of VMs, with the associated SLAs are known;
— Interference matrices are derived, following clients' submission, but prior to the VM placement process.

Also, this paper focuses on static VM placement problem as VM migration is not being addressed.

4.4.3 Model formulation

The model for the VM placement problem with the purpose of minimizing the total carbon footprint of an InterCloud environment is presented in the following.

4.4.3.1 Notation

The notation used to describe the optimization problem is composed of sets, constant parameters, decision and temperature variables, as well as other optimization parameters. See Appendix A for the complete model.

Sets

- D: the set of data centers ;
- X: the set of chassis ;
- S: the set of servers ;
- E: the set of energy sources ;
- K: the set of resources (CPU, disk, memory).

Constant parameters

- n_x^F: the number of fans of chassis $x \in X$;
- η_x^0: the fan power coefficient of chassis $x \in X$;
- a_{sk}: the power consumption of server $s \in S$ with 100% utilization of resource $k \in K$;
- p_s^S: the idle power of server $s \in S$;
- p_x^X: the idle power of chassis $x \in X$;
- e_{ed}: the percentage of power due to energy source $e \in E$ in data center $d \in D$;
- G_e: the carbon footprint, in Kg per CO_2, of energy source $e \in E$;
- G_d^{TOT}: the greenness factor of data center $d \in D$;

Variables

- u_{sk}: the usage of server $s \in S$ in terms of resource $k \in K$, with $u_{sk} \in [0\% - 100\%]$;
- z_x^X: a 0 - 1 variable, such as $z_x^X = 1$ if chassis $x \in X$ is active, or 0 otherwise ;
- z_s^S: a 0 - 1 variable, such as $z_s^S = 1$ if server $s \in S$ is active, or 0 otherwise ;
- τ_d: the supplied temperature in data center $d \in D$, with $\tau_d \in [15°\,\text{Celsius} - 35°\,\text{Celsius}]$;
- $COP_d(\tau_d)$: the Coefficient of Performance (COP) of data center $d \in D$;
- $\rho_d(\tau_d)$: the PUE of data center $d \in D$;
- $P_x^F(\tau)$: the fan power consumption of chassis $x \in X$;
- P_s^S: the power consumption of server $s \in S$;
- $P_x^{CH}(\tau)$: the power consumption of chassis $x \in X$ at temperature τ ;
- $P_d^{IT}(\tau)$: the power consumption of the IT equipment in data center $d \in D$;
- $P_d^{CO}(\tau)$: the CRAC consumption of data center $d \in D$;
- $P_d(\tau_d)$: the power consumption of data center $d \in D$;

Other optimization parameters

— α^X_{sx}: a 0-1 parameter, such as $\alpha^X_{sx} = 1$ if server $s \in S$ belongs to chassis $x \in X$;
— α^D_{sd}: a 0-1 parameter, such as $\alpha^D_{sd} = 1$ if server $s \in S$ belongs to data center $d \in D$;
— β_{xd}: 0-1 parameter, such as $\beta_{xd} = 1$ if chassis $x \in X$ belongs to data center $d \in D$;

4.4.3.2 Power models

This section describes the power models of the fans, servers and chassis, and the cooling system, as these entities represent the main contributors of the total electricity bill of large data centers (Pelley *et al.*, 2009). Due to the absence of traffic, the network resources are neglected.

Fan power

The fans considered in this paper vary their power over a range of inlet temperatures. In (Lee, 2012), the authors show that the fan power increases to the third power of the chassis inlet temperature τ, and is given by:

$$P^F_x(\tau) = \eta^0_x \tau^3, \quad \forall x \in X \tag{4.1}$$

Server power

The linear model based on the processor/disk utilization suggested by Sharifi *et al.* (2012) and Lee et Zomaya (2012) has been considered in order to evaluate a server power consumption:

$$P^S_s = p^S_s + a_{sk_1} * u_{sk_1} + a_{sk_2} * u_{sk_2}, \quad \forall s \in S \tag{4.2}$$

Chassis power consumption

The total chassis power consumption, $P^{CH}_x(\tau)$, includes its idle power, p^X_x, along with the fans and servers' power dissipation:

$$P^{CH}_x(\tau) = p^X_x + n^F_x * P^F_x(\tau) + \sum_{s \in S} \alpha^X_{sx} * z^S_s * P^S_s, \quad \forall x \in X \tag{4.3}$$

Cooling system power

The power consumption of a cooling system depends on its COP (Moore *et al.*, 2005), and is given by:

$$P^{CO}_d(\tau) = P^{IT}_d(\tau) / \mathrm{COP}_d(\tau), \quad \forall d \in D \tag{4.4}$$

where $\mathrm{COP}_d(\tau)$ is given by the following (Mazzucco *et al.*, 2010):

$$\mathrm{COP}_d(\tau) = 0.0068 * \tau^2 + 0.0008 * \tau + 0.458, \quad \forall d \in D \tag{4.5}$$

4.4.3.3 The model

The carbon footprint minimization model depends on the data centers power consumption and their greenness factor. The IT equipment power consumption of a given data center is expressed as follows:

$$P_d^{IT}(\tau_d) = \sum_{x \in X} \beta_{xd} * z_x^X * P_x^{CH}(\tau_d), \quad \forall d \in D \tag{4.6}$$

And with the CRAC power consumption given by (4.4), the power consumption of a data center d is therefore:

$$P_d(\tau_d) = \rho_d(\tau_d) * P_d^{IT}(\tau_d), \quad \forall d \in D \tag{4.7}$$

where the data center's PUE, $\rho_d(\tau_d)$ is expressed below:

$$\rho_d(\tau_d) = 1 + 1/\mathrm{COP}_d(\tau_d), \quad \forall d \in D \tag{4.8}$$

Given the power-carbon conversion rate (Moghaddam *et al.*, 2011):

$$G_d^{TOT} = \sum_{e \in E} G_e * e_{ed}, \quad \forall d \in D \tag{4.9}$$

the carbon footprint minimization of the InterCloud environment is expressed as follows:

$$MIN \quad \sum_{d \in D} G_d^{TOT} * \rho_d(\tau_d) * P_d^{IT}(\tau_d) \tag{4.10}$$

Although increasing the indoor temperature reduces the cooling power needed, as stated in equation (4.4), the dynamic behavior of the fans can diminish the cooling system gains. Therefore, as stated in (4.10), it is recommended to address the problem as a whole in order to minimize the total carbon footprint of an InterCloud environment.

The model is subjected to the following constraints:

— The *placement constraints* impose that all the VMs must be placed while forbiding placing a given VM on multiple sites.

— The *capacity constraints* state that, for a given resource and a defined server, the total resource utilization on that server, should not exceed its capacity.

— The *server and chassis activation constraints* determine when a server or a chassis can be turned on.

— The *interference and performance degradation constraints* guide the VM consolidation process by preventing any type of interference or degradation.

— The *supplied temperature constraints* impose that the air temperature supplied by the cooling system must be within the range specified by the ASHRAE (ASHRAE, 2011), namely between 15° Celsius and 35° Celsius.
— The *chassis outlet temperature constraints* state that the air temperature exiting a chassis should not exceed the maximal outlet temperature of that chassis.
— The *non-negativity and integrity constraints* reduce the domain of variables to integers or positive real numbers.

4.5 Problem analysis and re-statement

This section presents an analysis of the objective function, followed by the problem re-statement.

4.5.1 Problem analysis

In an ideal scenario, temperature would be held at either 15°Celsius or 35°Celsius, and the VM placement problem would only consist in determining the best host for each VM. However, based on (4.10), it is not obvious to know if the minimum carbon footprint is obtained with such trivial temperatures. Therefore, the following analysis will help in determining whether or not it is of a crucial importance to calculate the optimal temperature leading to the minimum carbon footprint. For simplicity, the analysis will be done with one data center where the load is fixed.

First, the monotony of the objective function has been studied. By combining (4.3), (4.6), (4.7), (4.8) and (A.17), and with G_d^{TOT} being a constant, the carbon footprint in one data center can be formulated as follows:

$$C_d = E_d * \tau_d^3 + F_d + \frac{E_d * \tau_d^3 + F_d}{a * \tau_d^2 + b * \tau_d + c}, \quad \forall d \in D \tag{4.11}$$

where

$$E_d = \sum_{x \in X} \beta_{xd} * z_x^X * n_x^F * \eta_x^0, \quad \forall d \in D \tag{4.12}$$

$$F_d = \sum_{x \in X} \beta_{xd} * z_x^X * p_x^X + \sum_{s \in S} \alpha_{sd}^D * z_s^S * P_s^S, \quad \forall d \in D \tag{4.13}$$

and a, b and c are respectively defined as coefficients in (4.5). The first derivative $C_d^{'}$ has been computed and with a positive denominator, only the numerator is relevant to the monotony analysis. The numerator has then been plotted in Fig. 4.2, with different values of E_d and

F_d.

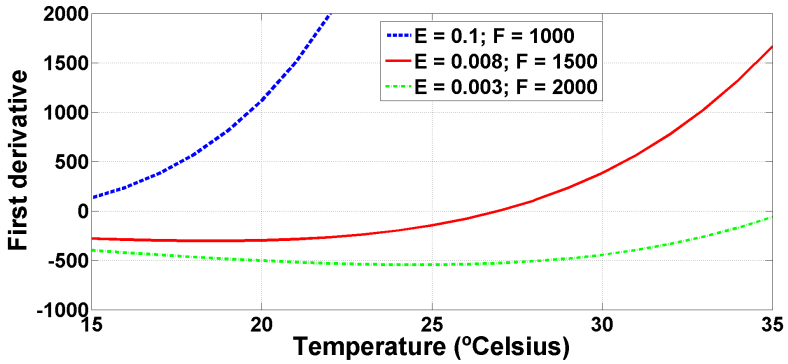


Figure 4.2 Monotony analysis

Considering a bounded search space, and a strictly monotonic function, this optimal temperature would be either 15°Celsius (increasing function) or 35°Celsius (decreasing function). However, from Fig. 4.2, it can be seen that the actual load largely influences the monotony of the function as the latter can be either strictly positive or negative or even change sign within the interval of interest. In that last scenario, it is hard to conclude whether an intermediate temperature value would lead to a minimum carbon footprint or not. Therefore, an analysis of the convexity has been conducted by computing the second derivative of (4.11), where the numerator (the denominator is always positive) is given by (due to space limitations, the calculation details are omitted):

$$\begin{aligned} num\left(C_d^{"}\right) &= \left(6a^2\tau_d^2 + 6ab\tau_d + 2b^2 - 2ac\right) F_d \\ &+ \left(18ac^2 - 2ac + 18b^2c + 2b^2\right) E_d\tau_d^3 + A \end{aligned} \tag{4.14}$$

where A represents the remaining positive terms of the second derivative of (4.11). Again, to analyze the sign of $C_d^{"}$ and determine the convexity of (4.11), the coefficients of E_d and F_d of (4.14) have been separately plotted on the interval of interest. The results are presented in Fig. 4.3.

As the two curves are always positive on the interval of interest I, it is reasonable to conclude that the function in (4.11) is always convex on I, regardless of the data center's load (E_d and F_d), and there is always an optimal temperature, namely 15°Celsius, 35°Celsius or any intermediate value, leading to the minimum carbon footprint. The problem analysis helped in showing that the crucial importance given to the determination of the optimal temperature value is not useless. Although considering a bounded search space and a convex objective

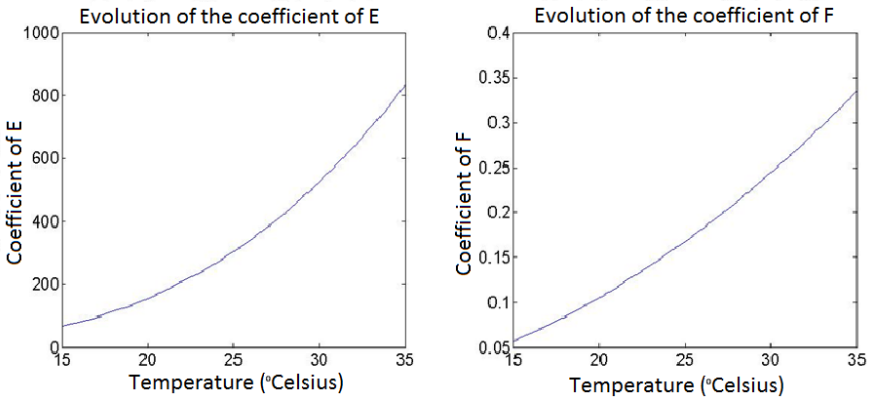


Figure 4.3 Evolution of the coefficient of E_d and F_d

function, the optimal temperature value leading to the minimum carbon footprint is not trivial to find, as the monotony of the function varies according to the load. Therefore, in the context of a carbon footprint minimization, the problem becomes more challenging, as an efficient VM placement should be combined with the evaluation of the best temperature, as the latter highly depends on the data center's load and characteristics.

4.5.2 Problem re-statement

Mathematical models based on a MINLP formulation are generally very difficult to solve. Therefore, it is important to reformulate the VM placement problem as a linear model in order to use ILP solver packages for obtaining the exact solution. Using the approach in (Pakbaznia et Pedram, 2009), we fix the supplied temperature of each data center to a constant value, τ_{d_FIX}, the outlet temperature of a chassis expressed by (A.7) is transformed into a linear expression, the COP of a data center becomes a constant and the objective function in (4.10) is reformulated as follows:

$$\begin{aligned} MIN \sum_{d \in D} B_d * \sum_{x \in X} \beta_{xd} * z_x^X * \left(p_x^X + n_x^F * \eta_{x\_FIX}^0 \right) \\ + \sum_{d \in D} B_d * \sum_{s \in S} \alpha_{sd}^D * z_s^S * P_s^S, \quad \forall d \in D \end{aligned} \tag{4.15}$$

with the constant fan power being expressed as:

$$\eta_{x\_FIX}^0 = \eta_x^0 * \tau_{d\_FIX}^3, \quad \forall x \in X, d \in D \tag{4.16}$$

Therefore, with a fixed value of the supplied temperature, the optimization problem in (4.15) has been formulated as a ILP model, where $z^X = \left[z_x^X \right]_X$ and $z^S = \left[z_s^S \right]_S$ are the unknown

variables, and with the nonlinear constraints being transformed into linear inequalities (see Appendix A for details). In order to minimize the total carbon footprint of the InterCloud, for each τ_{d_FIX} in the interval of interest, the ILP model presented in (4.15) will be solved using ILP solver packages. Solutions for each τ_{d_FIX} will be compared with each other, and the one with the minimum carbon footprint cost will determine the optimal temperature to be supplied for each active data center.

4.6 Computational experiments

In this section, experiments regarding the cost structure of the problem, as well as the analysis of the impact of the workload on the carbon cost, are conducted. For ease of illustration, a simplified version of the model, *CB_OPT*, where SLA requirements are neglected, is considered. Also, in order to demonstrate the contribution of the proposed model only in terms of carbon footprint minimization, *CB_OPT* is compared with other baseline approaches, where no SLA constraints are considered. Moreover, experiments conducted in Section 4.6.2.6 will help in assessing the good performance of the proposed model, *CB_OPT_SLA* compared to the modified version, *CB_OPT*, in terms of tradeoffs between the carbon footprint cost and the efficiency of the consolidation process. As the CPU time is also evaluated, starting from Section 4.6.2.3, all scenarios are run on a Core i3, 1.9 GHz CPU machine with 4 GB of RAM and running Windows 8.

4.6.1 Cost structure

The cost structure of a data center carbon footprint was studied in (Justafort *et al.*, 2014), where the equipment temperature-dependency was illustrated. Results showed that even in a single data center, the optimal solution is not easy to find, as the minimal carbon footprint can only be obtained within a temperature sweet spot that varies with the load and data center characteristics. The proposed model is therefore highly required to determine the optimal temperature, especially as such diligence should be exercised in each data center individually in an InterCloud environment.

4.6.2 Workload Analysis and Carbon Footprint Cost Comparison

In this section, the impact of the workload on the total carbon footprint, along with the efficiency of the proposed model with regard to other baseline approaches, in terms of total carbon footprint, are evaluated.

4.6.2.1 Experimentation setup

The model is implemented in AMPL, using the CPLEX solver. The features of the infrastructure and the VMs are presented in Table 4.2, with the number of clients set to 2. The sets of test instances are defined by the number of data centers D, chassis X, servers S and VMs V. Each set contains 3 instances, where the capacity and power consumption profile of each physical node and the resource requirements of each VM are randomly determined using a C++ program and the parameters in Table 4.2. In general, for a given instance, CPLEX tries to determine the best configuration, namely the VM-server assignment and the temperature to be supplied in each active data center, that minimizes the carbon footprint cost in (4.10). Once the optimal configuration is found, CPLEX returns the equivalent carbon footprint cost, computed with (4.10). To illustrate the results, the test instances are grouped by problem size, and metrics such as average carbon footprint cost, optimal temperature, CPU time and carbon footprint gap are provided. We also indicate, under the *Resource Limit (RL)* column, the percentage of test instances for which the optimal solution was not found, due to resource limitation.

4.6.2.2 Impact of the workload

Considering a single data center with 10 chassis and 100 servers, the effect of resource usage on the carbon footprint and the temperature has been studied. In the first scenario, the VMs among the sets are identical, while in the second scenario, the VMs are totally heterogeneous, even from a set to another. The results are presented in Table 4.3.

In the case of homogeneous VMs, as expected, the total carbon footprint increases along with the number of VMs. However, the same reasoning cannot be applied to the second situation, as a drop of carbon cost is observed for set 9. Due to the load heterogeneity, a growth in the number of VMs does not necessary lead to an increase of the carbon footprint. Regarding the temperature, although its raise is not tightly coupled with the increase of the IT carbon footprint, with high carbon cost, the CRAC unit needs to be set at high temperature values so the cooling gain negates the increase of the IT carbon footprint. As the impact of the workload, on the carbon footprint or the temperature, is generally unpredictable, the proposed model is highly required in order to find the optimal VM configuration and supplied temperature in the view of minimizing the total carbon footprint.

Table 4.2 Parameters

Data centers features	MIN	MAX
Greenness factor($KgCO_2/1000W$)	100	900
Number of chassis	4	12
Number of servers	20	250
Chassis features	MIN	MAX
Number of servers	5	25
Number of fans	6	10
Idle power	800	1000
Fan nominal power (W)	4	26
Maximum outlet temperature ($°C$)	93	100
Thermal resistance ($°C/W$)	0.03	1
Servers features	MIN	MAX
Processor capacity ($CPU - h$)	480	1200
Memory size (GBs)	500	1000
Disk capacity (GBs)	300	900
Idle power (W)	60	80
Processor power (W)	70	140
Disk power (W)	50	80
VMs features	MIN	MAX
Processor requirements ($CPU - h$)	12	24
Memory size (GBs)	5	20
Disk capacity (GBs)	5	15

Table 4.3 Impact of the workload

Instance set	Demand (V)	Homogeneous VMs			Heterogeneous VMs		
		Temp. ($°C$)	IT Carb. Foot. ($KgCO_2$)	Total Carb. Foot. ($KgCO_2$)	Temp. ($°C$)	IT Carb. Foot. ($KgCO_2$)	Total Carb. Foot. ($KgCO_2$)
1	20	19	1076.07	1443.58	19	1035.95	1389.76
2	40	20	1231.30	1616.80	19	1115.25	1496.15
3	60	20	1412.72	1855.03	20	1300.31	1707.41
4	80	21	1559.64	2008.64	20	1384.94	1818.55
5	100	21	1741.07	2242.30	20	1437.71	1887.84
6	120	22	1890.86	2392.84	21	1654.78	2131.16
7	140	22	2072.29	2622.43	21	1684.25	2169.13
8	160	23	2225.08	2771.30	22	1937.07	2451.32
9	180	23	2406.50	2997.26	22	1933.49	2446.79
10	200	23	2524.73	3144.51	23	2200.71	2740.95

4.6.2.3 Cost comparison in a unique data center

This simulation will assess the efficiency of *CB_OPT*, in the evaluation of a data center carbon footprint, with regards to three baseline methods. The first mechanism, the consolidation method (*CNS*), minimizes the number of active chassis and servers (Sharifi *et al.*, 2012), (Liu *et al.*, 2015). The second approach, *CB_MIN_TEMP*, optimizes the carbon footprint while considering the current average PUE (1.5) corresponding to the lowest temperature suggested by ASHRAE (2011) (15°Celsius) (Larumbe et Sanso, 2013). The third method, *CB_MAX_TEMP*, performs carbon footprint minimization while maximizing the CRAC unit efficiency by considering the maximal value of the temperature (35°Celsius) (Pakbaznia et Pedram, 2009). Different sets of tests, defined by the number of chassis, servers and VMs, are randomly generated, as the results for the *CB_OPT* model, regarding optimal temperature, carbon footprint costs, CPU time and percentage of unsolved set instances, are provided in Table 4.4. Similar results are shown in Table 4.5 for the reference models. However, the temperature are omitted and the carbon gaps instead of the carbon footprint are presented.

Table 4.4 Results for the proposed model *CB_OPT*

Instance set	Cloud size ($X-S-V$)	*CB_OPT*				
		Temp. ($°C$)	IT Carb. Foot. ($KgCO_2$)	Total Carb. Foot. ($KgCO_2$)	CPU time (s)	Proportion of RL (%)
1	10-50-100	20.67	1490.75	1931.99	45.53	0
2	10-100-200	22.33	2137.52	2690.32	445.06	0
3	10-150-300	23.33	2577.22	3193.53	3221.97	0
4	10-200-400	25.33	3192.60	3867.86	11552.05	33
5	10-250-500	—	—	—	28535.86	100

Table 4.5 Results for the baseline models

Instance set	*CNS*			*CB_MIN_TEMP*			*CB_MAX_TEMP*		
	Gap (%)	CPU time (s)	Proportion of RL (%)	Gap (%)	CPU time (s)	Proportion of RL (%)	Gap (%)	CPU time (s)	Proportion of RL (%)
1	17.22	1.81	0	5.74	2.20	0	23.40	4.32	0
2	22.65	1.78	0	8.52	16.26	0	15.50	14.74	0
3	28.41	4.45	0	9.96	74.95	0	12.11	282.51	0
4	30.91	13.53	0	12.67	1606.68	0	9.81	912.56	0
5	—	28.33	0	—	5955.96	66	—	4284.74	33

The exponential increase in the CPU time, observed in Table 4.4, is correlated to the problem size, namely the number of VMs and IT equipment. For the instances solved, the optimal temperature rises along with the total carbon cost, as explained in Section 4.6.2.2. As CPLEX uses Branch-and-Bound methods, when a large number of variables needs to be handled, the branch-and-cut tree increases and leads to insufficient memory. No optimal solution can therefore be found, as for instance sets 4 and 5. Also, based on the results in Table 4.5, *CB_OPT* outperforms the baseline approaches by performing workload consolidation and computing the optimal temperature. The cost gap between *CB_MIN_TEMP* and the optimal solution is sometimes smaller than the one observed with *CB_MAX_TEMP*, and vice versa, due to the fact that, in the first three scenarios, for example, the optimal temperature found by *CB_OPT*, is closer to the minimal value of 15°Celsius; while, in other cases, the supplied temperature is much higher. The continuous growth of the cost gap between the *CNS* method and the proposed model is mainly due to the fact that *CNS* model performs VM placement, with no consideration of the IT power characteristics. This latter approach is also the fastest one as it only minimizes the number of active equipment. The high complexity of the other models can be seen through the *RL* results, where at least 33% of certain instances are not solved. Nevertheless, *CB_OPT* is the most complex model to solve, as the CPU time can be 5 times higher than the execution time of the reference methods, and with a number of unsolved instance sets that can reach 100%, as in instance 5. Based on the results in Table 4.5, the proposed model can save up to one third of a cloud carbon footprint, mainly because the baseline methods lack an accurate objective function allowing them to perform an efficient carbon footprint minimization.

4.6.2.4 Impact of fan power consumption

The impact of the nominal power of a fan on data center optimal temperature and carbon footprint was analyzed in (Justafort *et al.*, 2014). Results showed that for a fixed infrastructure and load, the total carbon footprint increases along with the fan nominal power, while the optimal temperature is inversely proportional in the length of the input. As the latter grows, the IT carbon footprint increases faster than the decreasing of the cooling footprint, which allows the tradeoff point to be set at low temperature values. The simulations were extended to evaluate the *CB_OPT*'s savings compared to the *CB_MIN_TEMP* and the *CB_MAX_TEMP* models. Results revealed that these baseline models can only be used for low and high values of the nominal fan power consumption, with a maximal error margin of about 7.77%. However, as the carbon cost gap becomes more important for intermediate values of the fan power, these baseline models become no longer suitable. This simulation, again, helps in assessing the efficiency of the proposed approach as, regardless of the nominal

fan power, *CB_OPT* is able to determine the optimal configuration.

4.6.2.5 Cost comparison in an InterCloud environment

CB_OPT is then used to perform VM placement in an InterCloud in order to minimize the carbon footprint of such computing environment. The efficiency of the proposed model is compared with two reference methods: the first mechanism, *INS*, consolidates the VMs in clean clouds first (Moghaddam *et al.*, 2011), the other comparative model, *PWR*, minimizes the total power consumption of the InterCloud (Dong *et al.*, 2013). For that latter approach, the VM placement is first optimized for power consumption, and the resulting carbon footprint has then been evaluated. A network of 3 data centers characterized by different greenness factors is considered, and each set of test instance is defined by the number of chassis, servers and VMs. The carbon footprint, the CPU time and the cost gaps with the comparative models are provided in Table 4.6.

Table 4.6 Carbon footprint cost comparison for an InterCloud of 3 data centers

Instance set	InterC. size (X, S, V)	*CB_OPT*		*INS*			*PWR*		
		Carb. foot. ($KgCO_2$)	CPU time (s)	Carb. foot. ($KgCO_2$)	CPU time (s)	Gap (%)	Carb. foot. ($KgCO_2$)	CPU time (s)	Gap (%)
1	30-180-40	1032.39	804.48	1194.26	22.74	13.55	1165.29	3559.50	11.41
2	30-180-80	451.27	1616.39	480.63	42.74	6.11	959.73	3006.32	52.98
3	30-170-120	520.12	2181.73	526.85	48.94	1.28	744.83	3865.53	30.17
4	30-160-160	537.89	5022.35	551.45	51.41	2.46	537.89	8779.76	0
5	30-150-200	1793.16	3015.40	1804.59	69.45	0.63	2163.79	8510.48	17.13
6	30-180-240	677.19	3403.35	684.57	128.81	1.08	1952.14	7037.83	65.31
7	30-200-280	1061.43	205906	1082.95	157.90	1.99	1623.46	6575.90	34.62

The results show that *CB_OPT* always outperforms the other approaches in terms of carbon footprint cost. *INS* has the worst performance when the number of VMs is small compared to the number of servers: with more green options available, *INS* packs as many VMs as possible in the cleanest data centers, and ignores the IT equipment characteristics. As the load increases, due to the limited number of servers, *INS* becomes more constrained and its behavior tends to be similar to the proposed model. However, even in those cases, *INS* remains less efficient than the proposed model because it does not perform a power-aware VM consolidation. Regarding the power model, with a cost gap ranging from 0% to 65.31%, one can infer that *PWR* does not always achieve a good performance. Although it effectively selects power efficient hosts to place the VMs onto, it totally ignores the impact

of greenness factor during the VM placement process. Such results confirm the popular hypothesis stating that power optimization in an InterCloud is not always correlated to carbon footprint minimization in that same computing environment. Also, as observed in Table 4.6, *INS* is faster than the other models: since only the environmental impact of the data centers is important, *INS* sorts the data centers in ascending order of their greenness factor and pack the VMs accordingly. For the other models, not only the most power efficient equipment needs to be selected, but also all the temperature combinations should be tested in order to optimize the objective function. Also, it can be noted that the carbon footprint is strongly correlated to the hardware and VMs features as larger sets with more eco-friendly components may result in a lower carbon footprint cost compared to smaller problem sizes, as in instances 1 and 2, or 5 and 6. The results show that the proposed model can allow savings of up to 65% in terms of reduction in the carbon footprint cost.

4.6.2.6 Cost comparison with SLA constraints

This last scenario aims at analyzing the impact of VM placement on SLA violations and carbon footprint. *CB_OPT*, which neglects SLA constraints, is therefore compared to the proposed model *CB_OPT_SLA*, which includes the co-location requirements. Due to space limitation, results regarding other SLA constraints could not be presented. To assess the good quality of the *CB_OPT* approach and measure the merit of the consolidation process, we considered the CF coefficient in (Sharifi *et al.*, 2012) and evaluated as follows:

$$CF = \frac{Vi * C_{CB\_OPT\_SLA}}{Co * (C_{CB\_OPT\_SLA} - C_{CB\_OPT})} \tag{4.17}$$

where Co is the total number of SLA constraints and Vi is the number of violated ones, $C_{CB\_OPT\_SLA}$ and C_{CB_OPT} are respectively the carbon footprint cost obtained with *CB_OPT_SLA* and *CB_OPT*. For the purpose of that simulation, we consider an InterCloud of up to 3 data centers characterized by different greenness factors and where each set of test is defined by the number of data centers, chassis, servers and VMs. The results regarding the carbon footprint and the CPU time are provided in Table 4.7. The percentage of unsolved instances, when applied, is also presented (in parenthesis). The last three columns present the SLA constraints (Co), the number of violated ones (Vi) and the evaluation of the CF coefficient.

CB_OPT solves more instances than *CB_OPT_SLA*, because the latter introduces new variables that increase the computation complexity and results in memory problems. Due to that complexity, CPLEX is not able to solve large instances, such as sets 10, 11, 12 and 15,

Table 4.7 Results for carbon footprint and co-location constraints

Inst. set	InterC. size (D, X, S, V)	*CB_OPT_SLA*		*CB_OPT*				
		Carb. foot. ($KgCO_2$)	CPU time (s)	Carb. foot. ($KgCO_2$)	CPU time (s)	Constr. (Co) (#)	Violat. (Vi) (#)	CF
1	1, 4, 20, 20	1479.78	8.99	1400.52	8.14	73	73	18.67
2	1, 4, 20, 40	1581.72	29.21	1516.83	10.48	323	323	24.38
3	1, 4, 40, 40	1581.72	52.68	1516.83	13.92	323	323	24.38
4	1, 4, 40, 80	1830.77	573.28	1830.77	38.80	1189	621	Inf
5	1, 4, 60, 60	1704.16	2539.39	1680.26	49.71	634	408	45.88
6	1, 4, 60, 120	2131.16 (66)	27396.88	2111.83	131.51	2714	794	32.24
7	1, 8, 40, 40	1581.72	64.05	1516.83	13.65	323	323	24.38
8	1, 8, 40, 80	1830.77	599.80	1830.77	22.57	1189	596	Inf
9	1, 8, 80, 80	1830.77	2212.07	1830.77	46.54	1189	589	Inf
10	1, 8, 80, 160	*RL* (100)	8296.01	2434.07	327.72	4332	1155	—
11	1, 8, 120, 120	*RL* (100)	13927.61	2111.83	192.30	2714	906	—
12	1, 8, 120, 240	*RL* (100)	8296.01	2787.97	2774.72	11036	2170	—
13	1, 12, 60, 60	1602.65	274.55	1580.47	24.21	634	501	57.06
14	1,12, 60, 120	1952.07	8924.85	1930.41	74.10	2714	1061	35.22
15	1, 12, 120, 120	*RL* (100)	38917.47	1930.41	164.58	2714	1054	—
16	1, 12, 120, 240	*RL* (100)	*	2578.18	2129.38	11036	2035	—
17	1,12, 180, 180	*RL* (100)	519.70	2243.71	671.41	6240	1672	—
18	1, 12, 180, 360	*RL* (100)	*	3505.57 (66)	20424.39	23641	1062	—
19	2, 8, 40, 20	1096.78	88.01	1038.48	55.06	73	73	18.81
20	2, 8, 40, 40	1157.74	405.66	1099.44	66.26	323	323	19.86
21	2, 8, 80, 40	1157.74	788.43	1099.44	87.09	323	323	19.86
22	2, 8, 80, 80	1308.10	13861.82	1308.10	322.47	1189	523	Inf
23	2, 8, 120, 60	1231.34	7939.61	1231.34	252.90	634	351	Inf
24	2, 8, 120, 120	1504.33 (66)	50615.55	1524.69	1659.86	2714	594	-16.16
25	2, 16, 80, 40	1157.74	1065.81	1099.44	84.77	323	323	19.86
26	2, 16, 80, 80	1308.10	13122.51	1308.10	306.51	1189	554	Inf
27	2, 16, 160, 80	1308.10	35949.72	1308.10	506.84	1189	561	Inf
28	2, 16, 160, 160	*RL* (100)	*	1724.10	2755.16	4332	1001	—
29	2, 16, 240, 120	*RL* (100)	*	1524.69	2741.43	2714	619	—
30	2, 16, 240, 240	*RL* (100)	*	1965.01	21225.85	11036	2053	—
31	2, 24, 120, 60	192.96	5103.36	192.96	177.56	634	361	Inf
32	2, 24, 120, 120	236.83	29675.35	236.83	544.94	2714	670	Inf
33	2, 24, 240, 120	*RL* (100)	*	236.83	1060.86	2714	546	—
34	2, 24, 240, 240	*RL* (100)	*	310.02	7082.27	11036	2277	—
35	2, 24, 360 180	*RL* (100)	*	278.21	5104.97	6240	1296	—
36	2, 24, 360, 360	*RL* (100)	*	399.37 (33)	78695.34	23641	2154	—
37	3, 12, 60, 20	902.49	899.75	863.09	495.84	73	73	22.91
38	3, 12, 60, 40	983.46	5772.02	983.46	704.83	323	187	Inf
39	3, 12, 120, 40	983.46	9802.89	983.46	1113.96	323	175	Inf

and the corresponding CF value is not computed. For certain instance sets, such as 16, 18, 28, 29 and 30, *RL* is reached at preprocessing stage, therefore, no CPU time is evaluated. It can also be observed that *CB_OPT* is faster than *CB_OPT_SLA*, where more exploration

of the neighborhood is needed to find the best VM configuration, while considering the SLA requirements.

The carbon footprint obtained with *CB_OPT* is, on average, lower than the one given by *CB_OPT_SLA* because, in that latter case, more active servers may be required to efficiently span the VMs and prevent any type of interference or degradation. However, the outperformance of *CB_OPT* leads to non-negligible SLA violations, which percentage is significant, compared to the amount of saved carbon cost, which yields bad CF coefficients. The worst-case scenario is illustrated at scenarios 4, 8, 9, 22, where no carbon footprint improvement combined with non-negligible SLA violations are observed. The CF ratio, close to infinity, therefore depicts an extremely inefficient VM placement. The negative value of the CF coefficient at test instance 24, -16.16, indicates that *CB_OPT_SLA* yields more carbon savings. For that particular set, the average value of *CB_OPT*, evaluated on the 3 test instances, is higher than the results obtained with *CB_OPT_SLA*, for which only one third of the test instances have been solved. It can also be noted that for instances, like 2, 3 and 7, where the number of data centers and VMs are fixed, the carbon footprint resulting from the VM placement is identical, regardless of the number of chassis and servers, because the optimal configuration is already obtained with the smallest infrastructure size. However, the CPU time increases, on average, with the number of chassis and servers, mainly because more VM configurations need to be tested.

According to Sharifi *et al.* (2012), a CF less than 1 generally illustrates an efficient VM consolidation. Based on the results, where CF is at least 18.67 and may reach a value up to infinity, the SLA violations, introduced by *CB_OPT*, are extremely important compared to the carbon footprint saving gains. If these SLA violations are then converted into revenue loss or a penalty cost is associated to the GhG emissions, the cloud provider will have to evaluate whether it is beneficial or not to perform blind consolidation. Nevertheless, based on the results in Table 4.7, for instances like 9, 27, 32 and 39, the proposed model, *CB_OPT_SLA*, is able to determine the optimal VM configuration that leads to the minimal carbon footprint, while preventing any type of SLA violations from occurring.

4.7 Conclusion

In this paper, we proposed a new mathematical model, called *CB_OPT_SLA*, that minimizes the carbon footprint of an InterCloud environment. The results demonstrated that the VM placement process is not trivial, as the optimal carbon footprint value can only be obtained within a temperature sweet spot, and the impact of the workload on the optimal temperature or the environmental cost is unpredictable. The performance evaluation of the simplified

version of *CB_OPT_SLA*, along with the analysis of the impact of the nominal fan power, showed a significant carbon footprint reduction compared to other baseline methods. Other results also help in assessing the efficiency of *CB_OPT_SLA* regarding carbon footprint minimization while ensuring SLA requirements.

At this point, several research avenues are possible. As applications, spanning multiple VMs and leading to a non-negligible traffic, can be hosted in a cloud, the network power consumption can be included in the model. Also, techniques such as free cooling or heat redistribution can be considered in order to increase the CRAC efficiency. The proposed mathematical formulation can also be extended in order to adapt the model to online problems by integrating VM migration mechanisms. Furthermore, as the VM placement problem is classified as a NP-hard problem, it would be interesting to develop large-scale algorithms to find good feasible solutions in a reasonable amount of time.

Appendix A Model Formulation

A.1 Notation

A.1.1 Sets

Let D be the set of data centers and E, the set of energy sources. X is the set of chassis, S, the set of servers and K, the set of available resources (CPU, memory and disk). The set of clients and VMs are respectively C and V.

A.1.2 Power consumption parameters

Let p_x^X, R_x^0 and η_x^0 be respectively the idle power, the thermal resistance coefficient and the fan power coefficient of chassis $x \in X$. p_s^S is the idle power of server $s \in S$ and a_{sk}, the energy consumption of server $s \in S$ under 100% utilization of resource $k \in K$.

A.1.3 Other parameters

Let V_{vk_1}, V_{vk_2}, V_{vk_3} be respectively, the CPU, disk and memory requirements of VM $v \in V$. C_{sk_1}, C_{sk_2}, C_{sk_3} are the CPU, disk and memory capacity of server $s \in S$. Let $C_{sk_1}^M$ be the maximum CPU capacity allowed on server $s \in S$ to avoid performance degradation in the case of mixed workload and $C_{sk_2}^M$, the maximum disk capacity allowed to a VM on server $s \in S$ not to be considered as a disk-intensive VM. Let $I_{vv'}^V$ be the affinity coefficient between

VM v and VM v' ($v, v' \in V$), where $I^V_{vv'}$ is automatically null if v and v' belong to different clients, and $I^C_{vv'}$ the affinity coefficient between VM $v \in V$ and VM $v' \in V$, where $I^C_{vv'}$ is automatically null if v and v' belong to the same client. n^F_x is the number of fans of chassis $x \in X$. Let τ_{ref} be the reference temperature (15° Celsius), and $p^F_{x\_\tau_{ref}}$ be the fan power of chassis $x \in X$ at the reference temperature. $\tau^{out\_M}_x$ is the maximal outlet temperature allowed for chassis $x \in X$; τ^m_d and τ^M_d are respectively the minimal and maximal supplied air temperature allowed inside data center $d \in D$. Let G_e be the carbon footprint, in Kg per CO_2, of the energy source $e \in E$; e_{ed}, the percentage of power due to energy source $e \in E$ in data center $d \in D$ and G^{TOT}_d, the energy-carbon conversion rate for data center $d \in D$.

A.1.4 Decision variables

Let z^X_x be a 0-1 variable such that $z^X_x = 1$ if chassis $x \in X$ in active, z^S_s a 0-1 variable, such that $z^S_s = 1$ if server $s \in S$ is active and z^V_{vs} a 0-1 variable, such that $z^V_{vs} = 1$ if VM $v \in V$ is hosted on server $s \in S$. Let y_{vs} be a 0-1 variable, with $y_{vs} = 1$ if VM $v \in V$ hosted on server $s \in S$ is a disk-intensive VM and $y^{k_1}_s$ a 0-1 variable, where $y^{k_1}_s = 1$ if the CPU capacity used on server $s \in S$ exceeds the maximum capacity CPU $C^M_{sk_1}$ on that server. x^1_s is a 0-1 variable such that $x^1_s = 1$ if at least one disk-intensive VM is running on server $s \in S$, and x^2_s is a 0-1 variable where $x^2_s = 1$ if at least two VMs are running on server $s \in S$.

A.1.5 Other variables

Let $u_{sk_1}, u_{sk_2}, u_{sk_3}$ be, respectively, the CPU, disk and memory usage of server $s \in S$. n^S_s is the total number of VMs hosted on server $s \in S$. Let τ_d be the supplied temperature in data center $d \in D$ and $\tau^{out}_x(\tau_d)$, the outlet temperature of chassis $x \in X$ at temperature τ_d. $\mathrm{COP}_d(\tau_d)$ and $\rho_d(\tau_d)$ represent the COP and the PUE of data center $d \in D$. P^S_s is the power consumption of server $s \in S$, $P^F_x(\tau)$ and $P^{CH}_x(\tau)$ are respectively the fan power and the total power consumption of chassis $x \in X$ at temperature τ. Finally, $P^{IT}_d(\tau)$, $P^{CO}_d(\tau)$ and $P_d(\tau)$ are respectively the IT equipment power, the CRAC power and the total power consumption of data center $d \in D$.

A.1.6 Other optimization parameters

Let α^X_{sx} be a 0-1 parameter, such that $\alpha^X_{sx} = 1$ if server $s \in S$ belongs to chassis $x \in X$ and α^D_{sd}, a 0-1 parameter, where $\alpha^D_{sd} = 1$ if server $s \in S$ belongs to data center $d \in D$. β_{xd} is

0-1 parameter where $\beta_{xd} = 1$ if chassis $x \in X$ belongs to data center $d \in D$ and λ_{vc} is a 0-1 parameter with $\lambda_{vc} = 1$ if VM $v \in V$ belongs to client $c \in C$. Let $y_{vs}^{k_2}$ be a 0-1 parameter such as $y_{vs}^{k_2} = 1$ if the disk requirement of VM $v \in V$ exceeds the maximum disk capacity $C_{sk_2}^{M}$ allowed to a VM on server $s \in S$, not to be considered as a disk-intensive VM. $D_{k_1}^{M}$ and N are respectively the total CPU of all the servers and the number of VMs to be placed.

A.2 Power models

According to Pelley *et al.* (2009), in absence of traffic, the servers, chassis, fans and the cooling system are the main contributors of the electricity bill of large data centers.

A.2.1 Fan power

The fan power, with respect to the ambient temperature, is given by (Lee, 2012):

$$P_x^F(\tau) = \eta_x^0 * \tau^3, \quad \forall x \in X \tag{A.1}$$

with the fan power coefficient given by:

$$\eta_x^0 = p_{x\_t_{ref}}^F / \tau_{ref}^3, \quad \forall x \in X \tag{A.2}$$

A.2.2 Server power

The linear model suggested by Sharifi *et al.* (2012) and Lee et Zomaya (2012) is used to evaluate the server power consumption as follows:

$$P_s^S = p_s^S + a_{sk_1} * u_{sk_1} + a_{sk_2} * u_{sk_2}, \quad \forall s \in S \tag{A.3}$$

with the processor usage, for instance, expressed as follows:

$$u_{sk_1} = \sum_{v \in V} \left(z_{vs}^V * V_{vk_1} \right) / C_{sk_1}, \quad \forall s \in S \tag{A.4}$$

A.2.3 Chassis power consumption

The chassis power consumption includes its base power, along with the active servers and the fans power:

$$\begin{aligned} P_x^{CH}(\tau) =& p_x^X + n_x^F * P_x^F(\tau) \\ &+ \sum_{s \in S} \alpha_{sx}^X * z_s^S * P_s^S, \quad \forall x \in X \end{aligned} \tag{A.5}$$

A.2.4 Cooling system power

The cooling system power consumption is given by:

$$P_d^{CO}(\tau_d) = P_d^{IT}(\tau_d) / \mathrm{COP}_d(\tau_d), \quad \forall d \in D \tag{A.6}$$

with the COP curve quantified in terms of τ_d (Pakbaznia et Pedram, 2009).

A.3 Other parameters/variables computation

The outlet temperature of a chassis at temperature τ is:

$$\tau_x^{out} = P_x^{CH}(\tau) * \left(R_x^0/\tau\right) + \tau, \quad \forall x \in X \tag{A.7}$$

The number of VMs on a server is given by:

$$n_s^S = \sum_{v \in V} z_{vs}^V, \quad \forall s \in S \tag{A.8}$$

A.4 The cost function

The carbon footprint of an InterCloud environment depends on the data centers power consumption, namely the IT equipment power and the CRAC power dissipation, and their greenness factor. The IT equipment power consumption of a data center d is given by:

$$\begin{aligned} P_d^{IT}(\tau_d) =& \sum_{x \in X} \beta_{xd} * z_x^X * \left(p_x^X + n_x^F * P_x^F(\tau_d)\right) \\ &+ \sum_{s \in S} \alpha_{sd}^D * z_s^S * P_s^S, \quad \forall d \in D \end{aligned} \tag{A.9}$$

With the CRAC power given by equation (A.6), the total power consumption of a data center d is therefore:

$$P_d(\tau_d) = \rho_d(\tau) * P_d^{IT}(\tau_d), \quad \forall d \in D \tag{A.10}$$

where the data center's PUE is given by:

$$\rho_d(\tau) = 1 + 1/\mathrm{COP}_d(\tau_d), \quad \forall d \in D \tag{A.11}$$

The greenness factor is obtained as follows (Moghaddam *et al.*, 2012):

$$G_d^{TOT} = \sum_{e \in E} G_e * e_{ed}, \quad \forall d \in D \tag{A.12}$$

A.5 The model

The objective function describing the carbon footprint minimization is given by:

$$MIN \quad \sum_{d \in D} G_d^{TOT} * \rho_d(\tau_d) * P_d^{IT}(\tau_d) \tag{A.13}$$

The proposed model is subjected to the constraints presented in what follows:

All the VMs should be hosted, while each of them should be located on a unique server.

$$\sum_{s \in S} z_{vs}^V = 1, \quad \forall v \in V \tag{A.14}$$

The amount of resource of a given type used on a server should not exceed its capacity.

$$\sum_{v \in V} z_{vs}^V * V_{vk} \leq C_{sk}, \quad \forall k \in K, s \in S \tag{A.15}$$

Server and chassis activation constraints are given by:

$$z_s^S \geq z_{vs}^V, \quad \forall v \in V, s \in S \tag{A.16}$$

$$z_x^X \geq z_s^S * \alpha_{sx}^X, \quad \forall s \in S, x \in X \tag{A.17}$$

Interference constraints are expressed as follows:

$$-I_{vv'}^V * \left(z_{vs}^V * \lambda_{vc} + z_{v's}^V * \lambda_{v'c}\right) \leq 1, \quad \forall v, v' \in V, c \in C, s \in S \tag{A.18}$$

$$-I^C_{vv'} * \left[z^V_{vs} * (1 - \lambda_{vc}) + z^V_{v's} * \lambda_{v'c}\right] \leq 1, \quad \forall v, v' \in V, c \in C, s \in S \tag{A.19}$$

A server can host at most one disk-intensive VM (A.20)

$$\sum_{v \in V} y_{vs} \leq z^S_s, \quad \forall s \in S \tag{A.20}$$

with y_{vs} defined as in (A.21)

$$y_{vs} = y^{k_2}_{vs} * z^V_{vs}, \quad \forall v \in V, s \in S \tag{A.21}$$

Performance degradation constraints due to CPU over utilization is obtained as follows:

$$x^1_s + x^2_s + y^{k_1}_s \leq 2, \quad \forall s \in S \tag{A.22}$$

Equations (A.23) and (A.24) specify when there is a least one disk-intensive VM running on a server:

$$x^1_s \geq y_{vs}, \quad \forall v \in V, s \in S \tag{A.23}$$

$$x^1_s \leq \sum_{v \in V} y_{vs}, \quad \forall s \in S \tag{A.24}$$

Constraints (A.25) and (A.26) specify when there is at least two VMs running on a server:

$$2 - n^S_s \leq \left(1 - x^2_s\right) * N, \quad \forall s \in S \tag{A.25}$$

$$n^S_s \leq 1 + x^2_s * N, \quad \forall s \in S \tag{A.26}$$

Equations (A.27) and (A.28) state when the CPU capacity used exceeds the maximum allowed for a server:

$$C^M_{sk_1} - \sum_{v \in V} V_{vk_1} * z^V_{vs} \leq - \left(y^{k_1}_s/2\right) + \left(1 - y^{k_1}_s\right) * D^M_{k_1} \quad \forall s \in S \tag{A.27}$$

$$\sum_{v \in V} V_{vk_1} * z^V_{vs} \leq C^M_{sk_1} + y^{k_1}_s * D^M_{k_1}, \quad \forall s \in S \tag{A.28}$$

The supplied temperature should be in the range imposed by the ASHRAE (ASHRAE, 2011):

$$\tau_d \geq \tau^m_d, \quad \forall d \in D \tag{A.29}$$

$$\tau_d \leq \tau^M_d, \quad \forall d \in D \tag{A.30}$$

The air temperature exiting a chassis should not exceed its maximal outlet temperature:

$$\tau_x^{out\_M} \geq \tau_x^{out}(\tau), \quad \forall x \in X \tag{A.31}$$

Decision variables can only be Boolean while the non-negativity constraints reduce the domain of the other variables to integers or positive real numbers.

CHAPITRE 5 ARTICLE 2 : AN ITERATED LOCAL SEARCH APPROACH FOR CARBON FOOTPRINT OPTIMIZATION IN AN INTERCLOUD ENVIRONMENT

Auteurs : Valérie Danielle Justafort, Ronald Beaubrun et Samuel Pierre.
Revue : Accepté pour publication dans le journal *International Journal of Metaheuristics*, en Août 2015.

Abstract

In this paper, we address the problem of Virtual Machine (VM) placement in an InterCloud with regard to the reduction of the environmental impact of such environment. We propose a mathematical formulation based on a smart workload consolidation method and a cooling maximization technique that considers the dynamic behavior of the cooling fans. As the Virtual Machine Placement Problem (VMPP) is classified as an NP-hard problem, we propose an implementation of the Iterated Local Search (ILS) algorithm, *ILS_CBF*, in order to find good solutions in a reasonable time. Computational results allow to identify the parameters that reduce the carbon footprint costs. The comparison of the proposed heuristic with the exact method and other algorithms demonstrate that the obtained costs are relatively close to the lower bounds, ranging from 0% to a maximum distance less than 2.6%, and allow a good tradeoff between the quality of the solution and the computational time.

5.1 Introduction

In the last few years, the Information and Communication Technology (ICT) sector has witnessed the emergence of a new trend in the distributing computing community, the Cloud Computing (Zhu et Ammar, 2006)-(Chowdhury *et al.*, 2009). With the prevalence of the Infrastructure as a Service (IaaS) service model, one of the main challenges of such new paradigm is the Virtual Machine Placement Problem (VMPP) (Cardosa *et al.*, 2009)-(Hermenier *et al.*, 2009), which tends to optimally assign Virtual Machines (VMs) to servers.

Due to the intensive usage of large scale applications, data centers' power has dramatically grown and has attracted a great deal of attention (Greenpeace, 2010) and (Mazzucco *et al.*, 2010). As many aspects of the VMPP refer to well-known NP-hard problems, such as the knapsack or Bin-Packing Problem (BPP) (Wu, 2013), exact algorithms are unsuitable to solve moderate and large instances of the problem. Therefore, consolidation methods have

been implemented to minimize the number of active servers (Li *et al.*, 2009) or the servers' power consumption (Dupont *et al.*, 2012). As cooling power consumption accounts for about 30% (Pakbaznia et Pedram, 2009) of the total power consumed, various works have also focused on reducing the cooling power overhead and maximizing the data center efficiency (Moore *et al.*, 2005).

With the acknowledgment that Cloud carbon footprint is about 2% of Greenhouse Gas (GhG) emissions (Moghaddam *et al.*, 2012), and is constantly growing, the reduction of ICT sector environmental impact has lately become a great concern. However, in the view of minimizing the environmental impact of an InterCloud, the assignment problem becomes more challenging. As such computing environment is made of a mixture of data centers powered by different types of energy sources, an InterCloud's power consumption does not always reflect its carbon footprint (Moghaddam *et al.*, 2011). In particular, due to the heterogeneous nature of the physical infrastructure, intelligent methods need to be implemented for assigning the VMs to the most power-efficient equipment, while considering the data centers' power-to-carbon conversion rate or greenness factor (Moghaddam *et al.*, 2012) and (Moghaddam *et al.*, 2011), and not mechanically place the workload in the totally clean powered data centers first.

Furthermore, with the dynamic behavior of the IT cooling fans, increasing the temperature inside a data center, as suggested by a study from the American Society of Heating, Refrigerating and Air Conditioning Engineers (ASHRAE) (ASHRAE, 2011), does not always improve the data center efficiency, as the gain of the cooling system can be easily negated by the increase of the fan power consumption (Moss et Bean, 2009). Therefore, a Virtual Machine (VM) assignment process that ignores the equipment temperature dependency will not necessary lead to a minimum power consumption and carbon footprint.

Also, as early works have provided smart consolidation techniques to overcome the performance degradation problem (Sharifi *et al.*, 2012), other co-location constraints, such as redundancy and security, should also be considered in the VM consolidation process (Bonde, 2010).

In this paper, we are interested in a global approach to the VMPP. In order to address the challenges related to the complex process of minimizing an InterCloud carbon footprint, this paper jointly deals with different approaches.

— First, an accurate carbon footprint model, based on existing models, will be improved while taking into account the heterogeneous nature of the infrastructure and the dynamic behavior of the cooling fans. Also, techniques such as smart workload consolidation will be extended to take into account other consolidation constraints related to security or redundancy.

— Second, the VMPP will be stated as a Mixed-Integer Nonlinear Programming (MINLP) (Williams, 2013) problem in order to minimize the carbon footprint in an InterCloud environment;
— Third, although the formal methods used to solve Integer Linear Programming (ILP) problems provide optimal solutions, they can only scale up to small problems. Therefore, a local search heuristic with acceptable time complexity will be developed in order to obtain good feasible solutions for large problems.

This paper tackles the problem of minimizing the ecological impact of an InterCloud environment and is organized as follows. Section 5.2 discusses the relevant related works. The system environment is presented in Section 5.3, while the problem formulation is described in Section 5.4. Section 5.5 presents the adaptation of the Iterated Local Search (ILS) algorithm. Simulation results are illustrated in Section 5.6. Finally, conclusion and future works are presented in Section 5.7.

5.2 Related work

As the dissipation of a data center powered by semi or non-clean energy sources (Moghaddam *et al.*, 2011) could reflect its environmental impact (Lenzen, 2010), numerous efforts have been deployed to minimize the power consumption or the equivalent carbon footprint of cloud data centers (Moghaddam *et al.*, 2011).

Previous VM placement schemes, such as (Cardosa *et al.*, 2009), (Li *et al.*, 2009), (Feller *et al.*, 2011), (Buyya *et al.*, 2010) and (Dong *et al.*, 2013), have provided efficient tools for consolidating as many VMs as possible on the minimum number of servers in order to save energy. More specifically, Feller *et al.* (2011) defined the problem as an instance of the BPP and designed an algorithm based on the ant colony optimization metaheuristic for computing the VM placement. However, this approach is not suitable for large data centers, where the hypothesis of servers homogeneity does not always hold. Therefore, using the server power consumption introduced by Kansal *et al.* (2010), several works (Dupont *et al.*, 2012), (Larumbe et Sanso, 2012), (Larumbe et Sanso, 2013) and (Buyya *et al.*, 2009) have developed different approaches to place VMs with regards to the power consumption minimization.

Other works have also tried to reduce the power consumption while ensuring performance isolation (Sharifi *et al.*, 2012). More specifically, Kusic *et al.* (2009) maximized the provider's profit by minimizing both power consumption and Service Level Agreement (SLA) violations. In (Srikantaiah *et al.*, 2008), a VM placement algorithm based on energy-performance optimal points has been proposed, whereas in (Kim *et al.*, 2013), an interference model has been used to prevent SLA violations. However, other interference types related to security or

redundancy, can alter the Quality of Service (QoS) expressed in the SLA and should also be considered in the VM placement process.

Temperature-aware VM placement algorithms based on heat extracting methods have been introduced in (Patel *et al.*, 2003), (Sharma *et al.*, 2005), for reducing heat recirculation and minimizing the cooling power of a data center (Moore *et al.*, 2005). In this context, Pakbaznia et Pedram (2009) used the heat flow model presented by Tang *et al.* (2006) to minimize the server and cooling power cost of a data center. The dynamic behavior of the IT fans has been studied in (Moss et Bean, 2009) and, based on these findings, Lee (2012) presented a case study of the impact of data center inlet temperature on energy efficiency.

The carbon footprint minimization problem in an InterCloud was addressed by few researches (Van Heddeghem *et al.*, 2012), (Moghaddam *et al.*, 2011). In particular, Van Heddeghem *et al.* (2012) proposed a model based on the probability mass function of the binomial distribution which cannot accurately compute the carbon footprint of an InterCloud environment. To solve this issue, Moghaddam *et al.* (2012) and Moghaddam *et al.* (2011) proposed a mathematical model to optimize the total carbon footprint in a distributed cloud.

To the best of our knowledge, little prior work has been able to accurately formulate the optimization problem, by simultaneously considering the workload consolidation along with the SLA constraints, the cooling power dissipation, the dynamic behavior of the IT equipment and the cleanness of the data centers. As large instances of the VMPP will be considered, we will also provide an implementation of the ILS heuristic, whose solutions will be compared with lower bounds in order to assess the quality of the proposed method. The ILS metaheuristic has been chosen due to its simplicity and effective applicability regarding the resolution of hard problems (Congram *et al.*, 2002), allowing one to cope with performance and scalability issues when dealing with large instances of the VM placement problems (Leivadeas *et al.*, 2013).

5.3 System environment

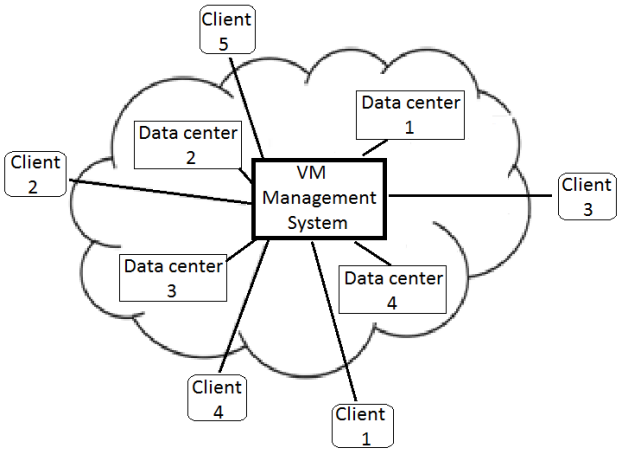
This section presents the main entities of the system environment, namely the InterCloud environment and the workload model.

5.3.1 InterCloud environment

The InterCloud, as illustrated in Figure 5.1, is made of three main components: a VM Management System used by the provider to optimize the VM placement, a set of data centers, and the clients with their applications. A data center is composed of racks of chassis

of multiple servers sharing fans and power supply. The servers are heterogeneous, provide physical resources in terms of CPU cycles, memory size and disk, and can run multiple VMs. Due to physical separators (Posladek, 2008), the chassis inlet temperature only depends on the cold air from the Computer Room Air Conditioner (CRAC) system, T_{sup}.

Figure 5.1 An InterCloud environment



5.3.2 Workload representation

Applications submitted by the clients are encapsulated in VMs that are characterized by a multidimensional resource vector and deployed among the servers. Due to technical constraints and for redundancy purposes (Bonde, 2010), VM interference constraints may require that certain VMs are not co-located. Therefore, based on the clients' requirements, in order to prevent interference between VMs of a client, or between VMs of different clients, interference matrices are then derived, as shown in Table 5.1. In such matrices, a negative value indicates an interference, and a null coefficient supposes no interaction. Moreover, as no performance degradation is allowed and based on the results in (Sharifi *et al.*, 2012), two VM placement constraints are defined: first, a server can host up to one disk-intensive application; second, the CPU utilization of a server should not exceed a threshold in the case

Table 5.1 Example of a VM-VM interference matrix

IDs	VM_1	VM_2	VM_3	VM_4
VM_1	0	0	-1	-1
VM_2	0	0	0	-1
VM_3	-1	0	0	0
VM_4	-1	-1	0	0

of mixed workload consolidation. Also, no inter-VM traffic is considered, as the VMs are running separately.

5.4 Problem statement

In this section, an overview of the VM placement process will be presented, followed by the mathematical model related to the optimization problem.

5.4.1 VM placement overview

As data centers are powered by different sources of energy, associating the carbon footprint minimization issue to the energy reduction problem, or packing the VMs in the environmentally friendly data centers first do not always yield the optimal solution. Furthermore, due to the equipment heterogeneity, VMs should be smartly assigned to the most power efficient hosts, while avoiding SLA violations. Moreover, in order to increase data centers' efficiency, balancing the fans' power consumption and the CRAC power dissipation through temperature control, is highly desirable. Based on these remarks, the carbon footprint minimization can only be realized by jointly 1) considering the greenness factor of the data centers; 2) performing workload consolidation, while respecting the SLA requirements; 3) determining the optimal value of the temperature inside each active data center. Also, Integer Programming (IP) techniques will be used to mathematically formulate the VM placement process.

5.4.2 Assumptions

In the context of the VM placement process, we make the following assumptions: the data centers, along with their equipment, are heterogeneous and initially set to off. We also consider a heterogeneous workload and integer values of the temperature varying between 15° and 35° Celsius, as suggested by ASHRAE (2011). Also, this paper focuses on static VM placement, as VM migration is not being addressed. Moreover, we assume that the data centers' features, such as the number and types of chassis and servers, along with the clients' requirements, in terms of number and types of VMs, with their SLA requirements, are known. We also assume that the interference matrices are derived, following the clients submission, but prior to the VM placement process.

5.4.3 Model formulation

The model for the VM placement problem with the purpose of minimizing the total carbon footprint of an InterCloud environment is presented in the following.

5.4.3.1 Notation

The notation used to describe the optimization problem is composed of sets, variables and parameters. See Appendix A for other variables or parameter computation.

Sets: Let D be the set of data centers and E the set of energy sources. X is the set of chassis, S, the set of servers, and K, the set of available resources (CPU, memory and disk). The set of clients and VMs are respectively C and V.

Power consumption parameters: Let p_x^X, R_x^0 and η_x^0 be respectively the idle power, the thermal resistance coefficient and the nominal fan power coefficient of chassis $x \in X$. p_s^S is the idle power of server $s \in S$ and a_{sk}, the energy consumption of server $s \in S$ under 100% utilization of resource $k \in K$.

Other parameters: Let V_{vk_1}, V_{vk_2}, V_{vk_3} be respectively the CPU, disk and memory requirements of VM $v \in V$. C_{sk_1}, C_{sk_2}, C_{sk_3} are the CPU, disk and memory capacity of server $s \in S$. Let $C_{sk_1}^M$ be the maximum CPU capacity allowed on server $s \in S$ to avoid performance degradation in the case of mixed workload, and $C_{sk_2}^M$, the maximum disk capacity allowed to a VM on server $s \in S$ not to be considered as a disk-intensive VM. Let $I_{vv'}^V$ be the affinity coefficient between VM v and VM v' $(v, v' \in V)$, where $I_{vv'}^V$ is automatically null if v and v' belong to different clients, and $I_{vv'}^C$ the affinity coefficient between VM $v \in V$ and VM $v' \in V$, where $I_{vv'}^C$ is automatically null if v and v' belong to the same client. n_x^F is the number of fans of chassis $x \in X$. Let $\tau_x^{out\_M}$ be the maximal outlet temperature allowed for chassis $x \in X$; τ_d^m and τ_d^M are respectively the minimal and maximal supplied air temperature allowed inside data center $d \in D$. Let G_e be the carbon footprint, in Kg per CO_2, of energy source $e \in E$; e_{ed}, the percentage of power due to energy source $e \in E$ in data center $d \in D$, and G_d^{TOT}, the energy-carbon conversion rate for data center $d \in D$.

Decision Variables: Let z_x^X be a 0-1 variable such as $z_x^X = 1$ if chassis $x \in X$ is active, z_s^S a 0-1 variable such as $z_s^S = 1$ if server $s \in S$ is active, and z_{vs}^V a 0-1 variable such as $z_{vs}^V = 1$ if VM $v \in V$ is hosted on server $s \in S$. Let $y_s^{k_1}$ be a 0-1 variable, where $y_s^{k_1} = 1$ if the CPU

capacity used on server $s \in S$ exceeds the maximum CPU capacity $C^M_{sk_1}$ on that server, and y_{vs} a 0-1 variable, with $y_{vs} = 1$ if VM $v \in V$ hosted on server $s \in S$ is a disk-intensive VM. x^1_s is a 0-1 variable such as $x^1_s = 1$ if at least one disk-intensive VM is running on server $s \in S$, and x^2_s is a 0-1 variable where $x^2_s = 1$ if at least two VMs are running on server $s \in S$.

Other variables: Let $u_{sk_1}, u_{sk_2}, u_{sk_3}$ be respectively the CPU usage, the disk usage and the memory usage of server $s \in S$. n^X_x and n^S_s are respectively the number of active servers of chassis $x \in X$ and the total number of VMs hosted on server $s \in S$. Let τ_d be the supplied air temperature in data center $d \in D$, $\tau^{out}_x(\tau_d)$ and $P^M_x(\tau_d)$, the outlet temperature and the corresponding maximal power consumption of chassis $x \in X$ at temperature τ_d.

Other optimization parameters: Let α^X_{sx} be a 0-1 parameter, such as $\alpha^X_{sx} = 1$ if server $s \in S$ belongs to chassis $x \in X$, and α^D_{sd} a 0-1 parameter, where $\alpha^D_{sd} = 1$ if server $s \in S$ belongs to data center $d \in D$. β_{xd} is a 0-1 parameter where $\beta_{xd} = 1$ if chassis $x \in X$ belongs to data center $d \in D$, and λ_{vc} is a 0-1 parameter with $\lambda_{vc} = 1$ if VM $v \in V$ belongs to client $c \in C$. Let $y^{k_2}_{vs}$ be a 0-1 parameter such as $y^{k_2}_{vs} = 1$ if the disk requirement of VM $v \in V$ exceeds the maximum disk capacity $C^M_{sk_2}$ allowed to a VM on server $s \in S$, not to be considered as a disk-intensive VM, $D^M_{k_1}$ and $D^M_{k_2}$ are respectively the total CPU and disk capacity of all the servers, and N is the number of VMs to be placed.

5.4.3.2 Power models

According to Pelley *et al.* (2009), the cooling system and the IT equipment, which includes the servers, chassis and fans, are the main contributors of the total electricity bill of large data centers. Due to the absence of traffic, the network resource consumption is neglected.

Fan power

The IT fans considered in this paper continuously vary their power over a range of inlet temperatures (Moss et Bean, 2009). Considering the fan law in (Lee, 2012), the fan power consumption $P^F_x(\tau)$ at temperature τ is given by:

$$P^F_x(\tau) = \eta^0_x * \tau^3, \quad \forall x \in X \tag{5.1}$$

Server power

The linear model suggested by Sharifi *et al.* (2012) and Lee et Zomaya (2012) is used in this

paper, and the server power dissipation P_s^S is obtained as follows:

$$P_s^S = p_s^S + a_{sk_1} * u_{sk_1} + a_{sk_2} * u_{sk_2}, \quad \forall s \in S \tag{5.2}$$

with the processor usage, for instance, given by:

$$u_{sk_1} = \sum_{v \in V} \left(z_{vs}^V * V_{vk_1} \right) / C_{sk_1}, \quad \forall s \in S \tag{5.3}$$

Chassis power consumption

The overall chassis power consumption $P_x^{CH}(\tau)$ includes its base power, along with the active servers and the associated fan power dissipation (Pakbaznia et Pedram, 2009):

$$P_x^{CH}(\tau) = p_x^X + n_x^F * P_x^F(\tau) + \sum_{s \in S} \alpha_{sx}^X * z_s^S * P_s^S, \quad \forall x \in X \tag{5.4}$$

Cooling system power

The cooling system power consumption of a data center d depends on its efficiency, described by its Coefficient of Performance (COP) (Moore *et al.*, 2005), and is given by:

$$P_d^{CO}(\tau_d) = P_d^{IT}(\tau_d) / \mathrm{COP}_d(\tau_d), \quad \forall d \in D \tag{5.5}$$

with the COP curve quantified in terms of the supplied temperature (Pakbaznia et Pedram, 2009).

5.4.3.3 The cost function

The proposed model aims at placing a set of VMs in an InterCloud environment in order to minimize its total carbon footprint, which depends on the data centers power consumption, namely the IT equipment power and the CRAC power dissipation, and their greenness factor. The IT equipment power consumption of a data center d, $P_d^{IT}(\tau_d)$, is given by:

$$P_d^{IT}(\tau_d) = \sum_{x \in X} \beta_{xd} * z_x^X * P_x^{CH}(\tau_d), \quad \forall d \in D \tag{5.6}$$

From (5.5), the total power consumption inside a data center $P_d(\tau_d)$ is given by:

$$P_d(\tau_d) = P_d^{IT}(\tau_d) * [1 + 1/\mathrm{COP}(\tau_d)], \quad \forall d \in D \tag{5.7}$$

where the Power Usage Effectiveness (PUE) of a data center $\rho_d(\tau_d)$ is given by:

$$\rho_d(\tau_d) = 1 + 1/\mathrm{COP}(\tau_d), \quad \forall d \in D \tag{5.8}$$

The power-carbon conversion rate depends on the energy sources and is given by (Moghaddam *et al.*, 2012):

$$G_d^{TOT} = \sum_{e \in E} G_e * e_{ed}, \quad \forall d \in D \tag{5.9}$$

5.4.3.4 The model

The objective function describing the carbon footprint minimization is given by:

$$MIN \quad \sum_{d \in D} G_d^{TOT} * \rho_d(\tau_d) * P_d^{IT}(\tau_d) \tag{5.10}$$

The proposed model is subjected to the following constraints:

All the VMs should be hosted, while each of them should be located on a unique server.

$$\sum_{s \in S} z_{vs}^V = 1, \quad \forall v \in V \tag{5.11}$$

The amount of resource of a given type used on a server should not exceed its capacity.

$$\sum_{v \in V} z_{vs}^V * V_{vk} \leq C_{sk}, \quad \forall k \in K, s \in S \tag{5.12}$$

Server and chassis activation constraints are presented in equations (5.13) and (5.14).

$$z_s^S \geq z_{vs}^V, \quad \forall v \in V, s \in S \tag{5.13}$$

$$z_x^X \geq z_s^S * \alpha_{sx}^X, \quad \forall s \in S, x \in X \tag{5.14}$$

Interference constraints regarding VMs of the same client are expressed as follows:

$$-I_{vv'}^V * \left(z_{vs}^V * \lambda_{vc} + z_{v's}^V * \lambda_{v'c}\right) \leq 1, \quad \forall v, v' \in V, c \in C, s \in S \tag{5.15}$$

Interference constraints between VMs of different clients are presented below:

$$-I_{vv'}^C * \left[z_{vs}^V * (1 - \lambda_{vc}) + z_{v's}^V * \lambda_{v'c}\right] \leq 1, \quad \forall v, v' \in V, c \in C, s \in S \tag{5.16}$$

Equations (5.17) states that a server can host at most one disk-intensive VM :

$$\sum_{v \in V} y_{vs} \leq z_s^S, \quad \forall s \in S \tag{5.17}$$

with y_{vs} defined as in (5.18):

$$y_{vs} = y_{vs}^{k_2} * z_{vs}^V, \quad \forall v \in V, s \in S \tag{5.18}$$

Performance degradation constraints due to CPU over utilization is given by:

$$x_s^1 + x_s^2 + y_s^{k_1} \leq 2, \quad \forall s \in S \tag{5.19}$$

Equations (5.20) and (5.21) specify when there is at least one disk-intensive VM on a server:

$$x_s^1 \geq y_{vs}, \quad \forall v \in V, s \in S \tag{5.20}$$

$$x_s^1 \leq \sum_{v \in V} y_{vs}, \quad \forall s \in S \tag{5.21}$$

Constraints (5.22) and (5.23) specify when there is at least two VMs running on a server:

$$2 - n_s^S \leq \left(1 - x_s^2\right) * N, \quad \forall s \in S \tag{5.22}$$

$$n_s^S \leq 1 + x_s^2 * N, \quad \forall s \in S \tag{5.23}$$

Equations (5.24) and (5.25) state when the CPU capacity used exceeds the maximum allowed:

$$C_{sk_1}^M - \sum_{v \in V} V_{vk_1} * z_{vs}^V \leq -\left(y_s^{k_1}/2\right) + \left(1 - y_s^{k_1}\right) * D_{k_1}^M, \quad \forall s \in S \tag{5.24}$$

$$\sum_{v \in V} V_{vk_1} * z_{vs}^V \leq C_{sk_1}^M + y_s^{k_1} * D_{k_1}^M, \quad \forall s \in S \tag{5.25}$$

The supplied temperature should be in the range imposed in (ASHRAE, 2011):

$$\tau_d \geq \tau_d^m, \quad \forall d \in D \tag{5.26}$$

$$\tau_d \leq \tau_d^M, \quad \forall d \in D \tag{5.27}$$

The temperature and power of a chassis should not exceed their threshold value:

$$\tau_x^{out\_M} \geq \tau_x^{out}(\tau), \quad \forall x \in X \tag{5.28}$$

$$P_x^{CH}(\tau) \leq P_x^{M}(\tau), \quad \forall x \in X \tag{5.29}$$

Decision variables can only be Boolean while the non-negativity constraints reduce the domain of the other variables to integers or positive real numbers.

5.5 The proposed ILS algorithm

In this section, we present the basic principles of the ILS heuristic, as well as the adaptation of such a method in order to efficiently solve the VM placement problem.

5.5.1 Basic principles

The ILS heuristic starts from an initial solution s_i, and iteratively applies appropriate moves to current solution s, in order to determine the best configuration. The Local Search (LS) process stops when there is no improvement of the best solution. Although fast, the LS mechanism may get trapped in a local minimum. For tackling this issue, a number of perturbations can be applied to the current solution in order to escape from the local optima.

5.5.2 Adaptation of ILS

In order to adapt the ILS algorithm to the problem, an initial solution, the LS process and perturbation mechanisms need to be implemented. Two values define a solution s: its actual cost $C(s)$, and its evaluation $E(s)$, which includes $C(s)$ and penalty cost $P(s)$, due to constraint violations. At each iteration, the LS algorithm accepts the best solution that decreases $E(s)$. Also, in order to avoid local optima, the proposed search method explores different perturbation techniques in order to obtain a good final solution.

5.5.2.1 Initial solution

In order to generate the initial solution, two methods are considered: *RandSol* and *ContrSol*. The first method randomly places the VMs on the servers and assigns a temperature to the data centers. In the second mechanism, data centers' temperature are set at 15° Celsius, then the VMs are listed in a descending order of their total resource requirements, and the

servers are sorted such as the ones in the green data centers are on top of the list. The initial solution is obtained by assigning each VM to the first suitable server.

5.5.2.2 LS algorithm

Based on an iterated descent algorithm, the LS mechanism improves the actual solution by executing a number of moves within its neighborhood in order to generate new solutions. For that purpose, the mechanism uses different types of moves defined as follows:

— $M_1(v, s_1, s_2)$: move VM v from server s_1 to server s_2 ;

— $M_2(d, t_1, t_2)$: change the temperature of data center d, from t_1 to t_2.

To rapidly evaluate the solutions in the neighborhood of the current one, two gains are defined: $G_1(v, s_1, s_2)$ and $G_2(d, t_1, t_2)$. The gain, $G_1(v, s_1, s_2)$, associated to the first move is given by:

$$G_1(v, s_1, s_2) = \begin{cases} M & \text{if} \quad s_1 = s_2 \\ G_1^C + G_1^O + G_1^T + G_1^D + G_1^I & \text{otherwise} \end{cases} \tag{5.30}$$

Although the gain is null when $s_1 = s_2$, a large value M is temporary assigned to the gain when $s_1 = s_2$ to avoid cycling on the same solution. Let d_1, τ_1 and x_1 be respectively the data center, the inlet temperature and the host chassis of server s_1. d_2, τ_2 and x_2 are respectively the host data center, the inlet temperature and the host chassis of server s_2. The carbon cost difference, G_1^C is expressed as follows:

$$G_1^C = G_{d_2}^{TOT} * \rho_{d_2}(\tau_2) * P_{vs_2}^{\Delta+}(\tau_2) - G_{d_1}^{TOT} * \rho_{d_1}(\tau_1) * P_{vs_1}^{\Delta-}(\tau_1) \tag{5.31}$$

with the increase in power consumption due to VM v being expressed by:

$$\begin{aligned} P_{vs_2}^{\Delta+}(\tau_2) &= (a_{s_2k_1} * V_{vk_1}/C_{s_2k_1}) + (a_{s_2k_2} * V_{vk_2}/C_{s_2k_2}) + \left(1 - z_{s_2}^S\right) p_{s_2}^S \\ &+ \left(1 - z_{x_2}^X\right) * \left(p_{x_2}^X + n_{x_2}^F * P_{x_2}^F(\tau_2)\right) \end{aligned} \tag{5.32}$$

and the decrease in power consumption due to VM v given by the following:

$$\begin{aligned} P_{vs_1}^{\Delta-}(\tau_1) &= p_{s_1}^S * \left(2 - n_{s_1}^S\right)^+ + (a_{s_1k_1} * V_{vk_1}/C_{s_1k_1}) + (a_{s_1k_2} * V_{vk_2}/C_{s_1k_2}) \\ &+ \left(2 - n_{x_1}^X\right)^+ * \left(1 - \alpha_{s_2x_1}^X\right) * \left(p_{x_1}^X + n_{x_1}^F * P_{x_1}^F(\tau_1)\right) \end{aligned} \tag{5.33}$$

The penalty gain, G_1^O, due to server over-utilization is given by:

$$G_1^O = \sum_{k \in K} \Psi^N(V_{vk}, C_{s_2k}, u_{s_2k} * C_{s_2k}) - \sum_{k \in K} \Psi^O(V_{vk}, C_{s_1k}, u_{s_1k} * C_{s_1k}) \tag{5.34}$$

with the generic functions $\Psi^N(W,C,Q)$ and $\Psi^O(W,C,Q)$ defined as follows:

$$\Psi^N(W,C,Q) = \begin{cases} \Omega + \omega * (W - C + Q) & \text{if} \quad Q \leq C, W > C - Q \\ \omega * W & \text{if} \quad Q > C \\ 0 & \text{otherwise} \end{cases} \tag{5.35}$$

$$\Psi^O(W,C,Q) = \begin{cases} \Omega + \omega * (W + C - Q) & \text{if} \quad Q > C, W \geq Q - C \\ \omega * W & \text{if} \quad Q > C, W < Q - C \\ 0 & \text{otherwise} \end{cases} \tag{5.36}$$

where Ω and ω are parameters to be determined experimentally.
G_1^T represents the penalty cost related to the chassis maximal power consumption:

$$\begin{aligned} G_1^T &= \Psi^N\Big(P_{vs_2}^{\Delta+}(\tau_2), P_{x_2}^M(\tau_2), P_{x_2}^{CH}(\tau_2)\Big) \\ &- \Psi^O\Big(P_{vs_1}^{\Delta-}(\tau_1), P_{x_1}^M(\tau_1), P_{x_1}^{CH}(\tau_1)\Big) \end{aligned} \tag{5.37}$$

where $P_x^M(\tau)$ is given by equation (A.2).

The penalty cost due to performance degradation is expressed as follows:

$$\begin{aligned} G_1^D = \Omega * \Bigg(& x_{s_2}^1 * y_{vs_2}^{k_2} - y_{vs_1} * \Bigg(\Bigg(\sum_{v' \in V} y_{v's_1} - 1 \Bigg)^+ \\ & - \Bigg(\sum_{v' \in V} y_{v's_1} - 2 \Bigg)^+ \Bigg) \Bigg) - x_{s_1}^1 * x_{s_1}^2 * \Psi^O\left(V_{vk_1}, C_{s_1k_1}^M, u_{s_1k_1} * C_{s_1k_1}\right) \\ & + x_{s_2}^1 * \Psi^N\left(V_{vk_1}, C_{s_2k_1}^M, u_{s_2k_1} * C_{s_2k_1}\right) + z_{s_2}^S * y_{vs_2}^{k_2} * \left(1 - x_{s_2}^1\right) * \Upsilon(v, k_1, s_2) \\ & - x_{s_1}^1 * x_{s_1}^2 * y_{vs_1} * \Bigg(2 - \sum_{v' \in V} y_{v's_1} \Bigg)^+ * \Upsilon(v, k_1, s_1) \\ & + x_{s_1}^1 * x_{s_1}^2 * y_{vs_1} * \Bigg(2 - \sum_{v' \in V} y_{v's_1} \Bigg)^+ * \Psi^O\left(V_{vk_1}, C_{s_1k_1}^M, u_{s_1k_1} * C_{s_1k_1}\right) \end{aligned} \tag{5.38}$$

The generic function $\Upsilon(v,k,s)$ is given by:

$$\begin{aligned} \Upsilon(v,k,s) = & \Big[\Omega + \omega * \Big(V_{vk} - \Big(C_{sk}^M - u_{sk} * C_{sk} \Big) \Big) \Big] \\ & * \Big[\Big(V_{vk} - \Big(C_{sk}^M - u_{sk} * C_{sk} \Big) \Big)^+ - \Big(V_{vk} - \Big(C_{sk}^M - u_{sk} * C_{sk} \Big) - 1 \Big)^+ \Big] \end{aligned} \tag{5.39}$$

Let c_1 be the owner of VM v. The penalty cost due to interference is expressed as follows:

$$G_1^I = \Omega \sum_{\substack{v' \in V \\ v' \neq v}} \left(z_{v's_2}^V - z_{v's_1}^V \right) * \left[(1 - \lambda_{v'c_1}) * \left(-I_{vv'}^C \right) + \lambda_{v'c_1} * \left(-I_{vv'}^V \right) \right] \tag{5.40}$$

The overall gain related to the second move type is given by:

$$G_2(d, t_1, t_2) = \begin{cases} M & \text{if} \quad t_1 = t_2 \\ G_2^C + G_2^T & \text{otherwise} \end{cases} \tag{5.41}$$

where a large value M is temporary assigned to $G_2(d, t_1, t_2)$ when $t_1 = t_2$ to avoid cycles. G_2^C is the carbon footprint cost difference when the supplied temperature inside data center d is changed from t_1 to t_2.

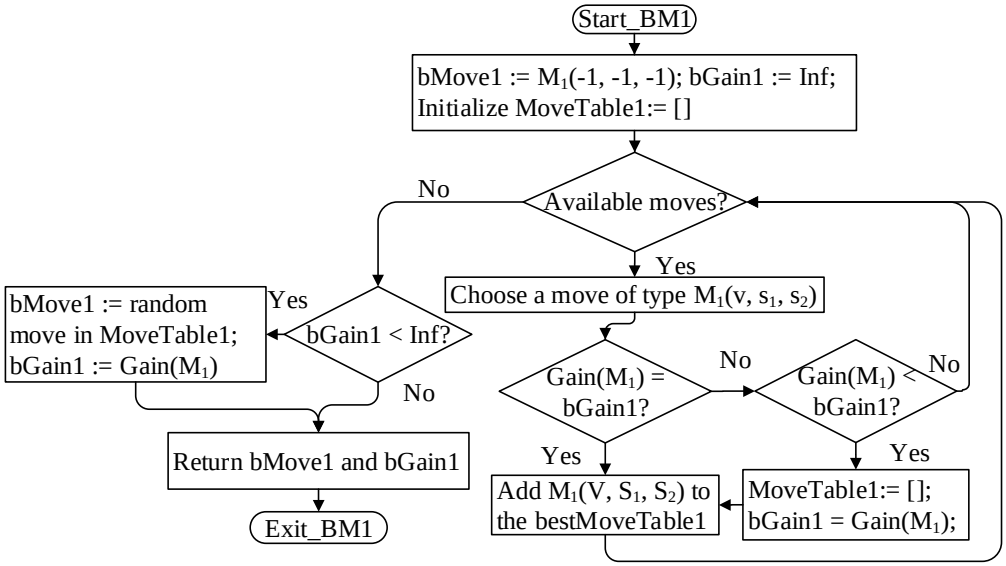
$$\begin{aligned} G_2^C = & \left[\sum_{x \in X} z_x^X * \beta_{xd} * \left(\rho_d(t_2) * \left(p_x^X + n_x^F * P_x^F(t_2) \right) \right. \right. \\ & \left. - \rho_d(t_1) * \left(p_x^X + n_x^F * P_x^F(t_1) \right) \right) \\ & \left. + \sum_{s \in S} z_s^S * \alpha_{sd}^D * P_s^S * \left(\rho_d(t_2) - \rho_d(t_1) \right) \right] * G_d^{TOT} \end{aligned} \tag{5.42}$$

The penalty gain associated to a chassis outlet temperature, G_2^T, is given by:

$$\begin{aligned} G_2^T = & \sum_{x \in X} \beta_{xd} * z_x^X * \left[\left((t_2 - t_1)^+ - (t_2 - t_1 - 1)^+ \right) \right. \\ & * \Psi^N \left(T_x^\Delta, \tau_x^{out\_M}, \tau_x^{out} \right) - \Psi^O \left(T_x^\Delta, \tau_x^{out\_M}, \tau_x^{out} \right) \\ & \left. * \left((t_1 - t_2)^+ - (t_1 - t_2 - 1)^+ \right) \right] \end{aligned} \tag{5.43}$$

with the outlet air temperature difference of chassis x, T_x^Δ, given by equation (A.3). Also, as presented in Figure 5.2 which depicts the approach used to select the best move for type 1, instead of accepting the first move with the lowest gain, the algorithm considers all the moves with the minimum gain, and randomly chooses one of them. Once the best move for each type has been determined, the descent algorithm chooses the one that yields the weakest gain. Such best move is then applied to the current solution. Gain tables are then updated, and the current solution is added to the statistics table, which keeps track of the number of times a VM is placed on a server, and a temperature value is set in a data center. The exploration process, presented in Figure 5.3, stops when there is no improved neighbors of

Figure 5.2 Best move algorithm



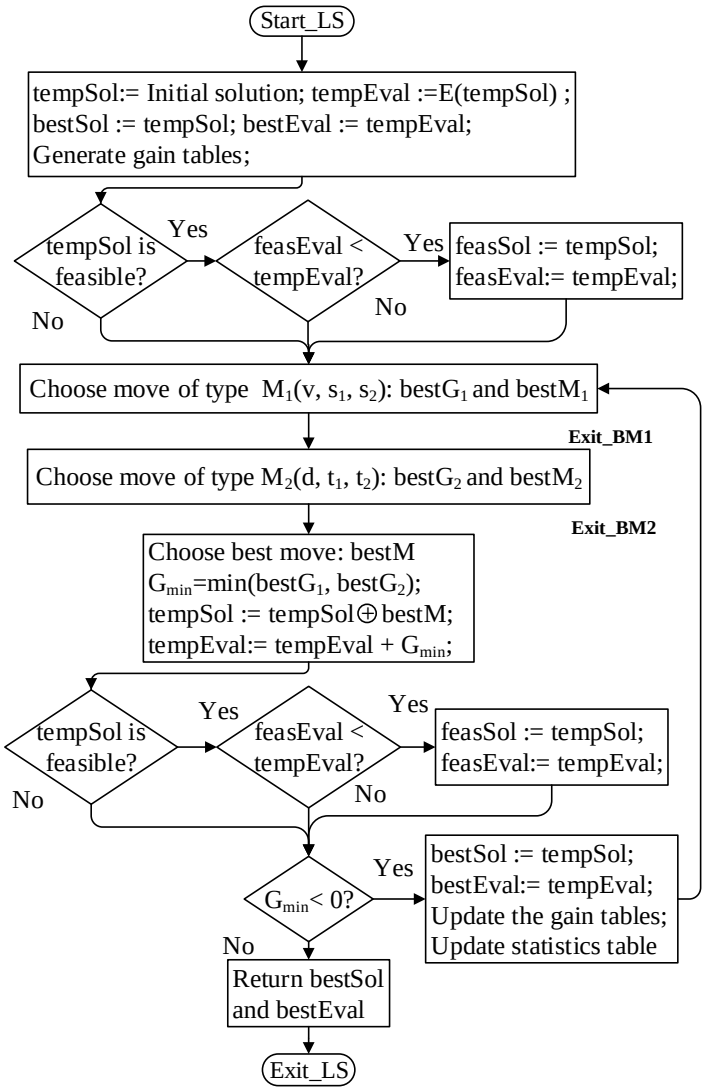
the actual solution.

5.5.2.3 Perturbation

In order to avoid being trapped in a local optimum, the ILS heuristic perturbs the solution obtained by the descent algorithm, and restarts the LS process using a different initial configuration. In order to implement the perturbation mechanism, four key parameters are defined: the type of perturbation ($pType$), its strength ($pStr$), the number of restarts ($nbST$) and the acceptance criterion. $nbST$ is the number of iterative calls to the LS process, $pStr$ refers to the percentage of a solution components altered by the perturbation and the proposed acceptance criterion imposes to only accept improved solutions. Figure 5.4 presents an overview of the perturbation mechanism.

Regarding the perturbations, three modes, $pMode_1$, $pMode_2$ and $pMode_3$ are examined. In the first mode, a unique type of perturbation ($pType$), among the three methods considered in this work, is used for $nbST$ calls of the LS algorithm. The first approach, *Random Restart*, restarts the LS process from a randomly generated configuration; the second method, *Swapping*, consists of swapping VMs hosted on different servers; and the third type, *New Host*, assigns VMs to unused servers. In the second mode, the algorithm randomly selects one of the perturbation types to generate the new solution. The last mode $pMode_3$ is $pMode_1$ with a callback mechanism, which is triggered every time the best solution has not been improved for

Figure 5.3 Descent algorithm

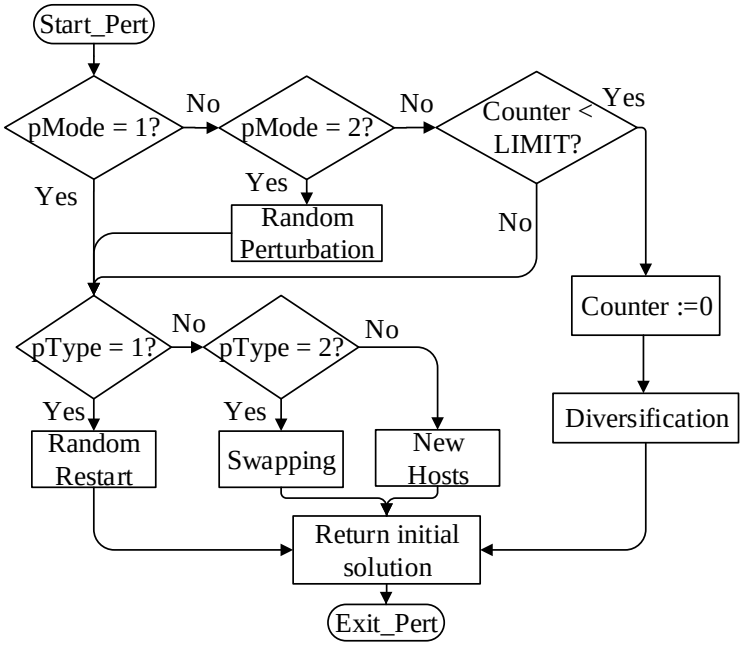


LIMIT consecutive iterations. The callback mechanism refers to the *Diversification*, which generates a new solution by considering the least explored placement pattern. Figure 5.5 illustrates the general steps of the ILS algorithm.

5.6 Computational experiments

In this section, the efficiency of the proposed ILS algorithm is evaluated through extensive simulations. Two sets of experiments are conducted: the first one is designed to determine the optimal values of the ILS parameters; the second set of experiments is intended to assess

Figure 5.4 Perturbation mechanism



the performance of the ILS method. The proposed algorithm is implemented in C++ under Visual Studio 12.1. Different input data files are also generated, using simulation parameters in Table 5.2 and a developed C++ program. All simulations are run on a Core i3, 1.9 GHz CPU Machine with 4 GB of RAM and running Windows 8.

5.6.1 ILS Implementation

This section presents the results obtained from implementing each mechanism of the ILS algorithm, with the purpose of determining the optimal key parameters leading to the best performance of the proposed heuristic.Therefore, simulations are run for each instance defined by a fixed physical substrate, namely 3 data centers, 30 chassis and 300 servers, and an incremental number of VMs (300-2100 VMs with a step of 300 VMs). For sake of simplicity and without loss of generality, the penalty cost due to performance degradation and chassis outlet temperature are not implemented. Regarding the VMs compatibility, no co-location constraints are considered in *Scenario 1*, while in *Scenario 2*, the probability of VM incompatibility is set to 0.5. Each problem size is executed 10 times using the ILS method, while metrics, such as average carbon footprint cost and CPU time are provided. Due to space limitation, only results from *Scenario 1* are presented.

Figure 5.5 ILS algorithm

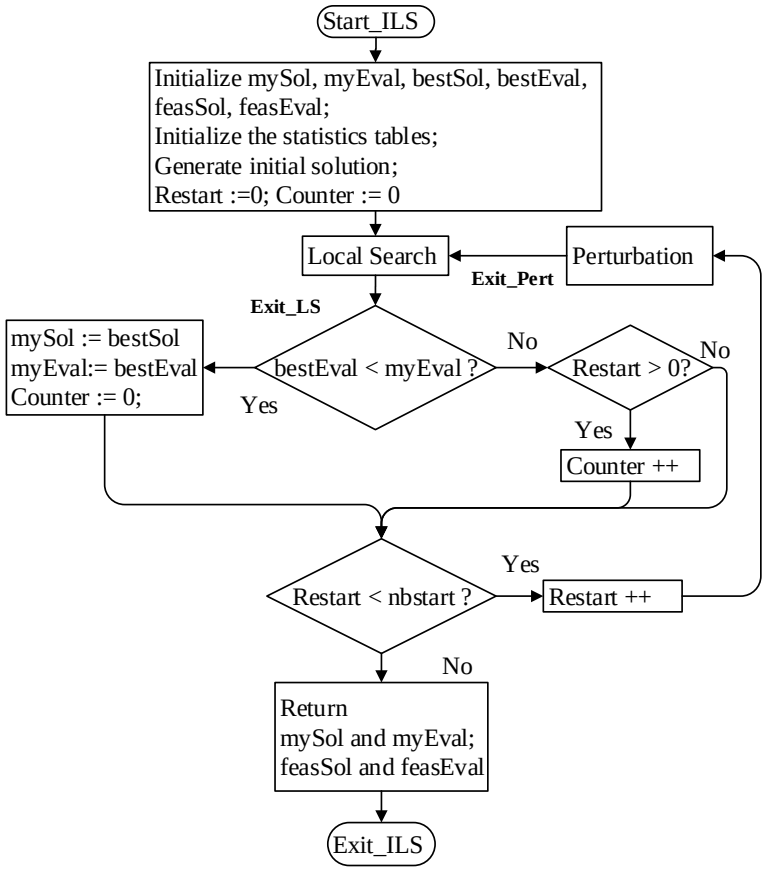


Table 5.2 InterCloud Features

Data centers features	MIN	MAX	VMs features	MIN	MAX
Greenness factor($KgCO_2/KW$)	100	900	Processor ($CPU - h$)	25	100
Number of chassis	4	12	Memory size (GBs)	25	100
Number of servers	20	250	Disk capacity (GBs)	25	100
Chassis features	MIN	MAX	Servers features	MIN	MAX
Number of servers	5	25	Processor capacity ($CPU - h$)	500	1000
Number of fans	6	12	Memory size (GBs)	500	1000
Idle power	500	1000	Disk capacity (GBs)	500	1000
Fan nominal power (W)	5	12	Idle power (W)	60	100
Maximum outlet temperature ($°C$)	93	100	Processor power (W)	70	240
Thermal resistance ($°C/W$)	0.03	1	Disk power (W)	50	280

5.6.1.1 Initial Solution and LS Implementation

In order to assess the impact of the initial configuration and the penalty cost on the solution, the proposed algorithm is evaluated under the approaches presented in Section 5.5.2.1 and

Table 5.3 *Scenario 1*: Impact of Ω, ω and the initial solution on the carbon footprint cost

Inst. set	*RandSol*						*ContrSol*					
	500/500		1000/1000		2000/2000		500/500		1000/1000		2000/2000	
	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)
1	5.95	0.97	5.95	0.99	5.95	0.96	3.89	0.04	3.89	0.03	3.89	0.03
2	12.13	2.46	12.13	2.43	12.13	2.42	8.58	0.10	8.58	0.10	8.58	0.11
3	15.17	4.75	15.16	4.78	15.16	4.78	12.82	1.30	12.82	1.30	12.82	1.32
4	17.68	7.95	17.68	8.15	17.68	7.91	16.60	0.23	16.60	0.23	16.60	0.23
5	21.81	11.58	21.81	11.59	21.81	11.78	23.94	0.92	23.94	0.88	23.94	0.88
6	25.90	13.83	25.90	13.88	25.90	14.02	27.66	1.16	27.66	1.16	27.66	1.17
7	48.28	12.69	48.31	12.74	48.16	12.64	48.13	0.21	48.13	0.21	48.13	0.20

with $\Omega = \omega = 500, 1000, 2000$. Results in Table 5.3 show that *ContrSol* provides on average the best solution, both in terms of carbon footprint (in tons of CO_2), and CPU time (in seconds). This is mainly due to the fact that, as the initial solution provided by *ContrSol* is a feasible and low-cost configuration, it becomes easier to explore the search space more efficiently and obtain a good local minimum instead of starting from a random point. Moreover, although the penalty coefficients do not affect the carbon cost in the case of *ContrSol*, they need to be properly sized in order to avoid accepting too many unfeasible solutions or penalizing promising ones, which may increase the CPU time. As the best values of Ω and ω are generally obtained with 1000, for the remaining simulations, the initial configuration will be generated using *ContrSol*, with the coefficients $\Omega = \omega = 1000$.

5.6.1.2 Perturbation Implementation - Simple Mechanisms ($pMode_1$)

This simulation aims at evaluating the impact of the perturbation process on the solution when simple perturbation mechanisms are used ($pMode_1$), which will enable to determine the optimal values of ILS parameters. As a first step, the algorithm is run under the three perturbation types, while keeping *pStr* and *nbST* constant. Results in Table 5.4 show that the *Swapping* method generally outperforms the other mechanisms. This is due to the fact that, with *Random Restart*, as each new configuration is randomly generated, no guarantee can be made regarding the quality of the new initial solution, while the *New Hosts* mechanism alters the best solution by using components that are usually not part of the actual best solution, which may lead to poor initial configurations.

In order to assess the impact of the perturbation strength (*pStr*) on the solution, the heuristic is executed using the *Swapping* method with $pStr = 25\%, 50\%, 75\%, 80\%, 90\%$ and *nbST* being kept fixed. From Table 5.5, it can be seen that the worst solutions are obtained with

Table 5.4 *Scenario 1*: Impact of the perturbation mechanism *pType* on the carbon footprint cost

Inst. set	Perturbation mechanisms (*pType*)					
	Random Restart		*Swapping*		*New Hosts*	
	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)
1	3.2723	20.0347	3.7844	1.8662	3.2723	20.5034
2	8.5762	43.5507	8.0733	3.7182	8.5628	24.0058
3	12.8159	69.6729	12.3619	8.7450	12.8159	29.3396
4	16.6024	110.7891	14.5411	11.0689	16.6024	29.7871
5	21.4920	150.7159	20.4867	17.7381	21.8381	34.4434
6	25.8125	185.3964	25.7853	20.2578	26.0958	32.2603
7	44.4345	200.0898	36.8415	26.9922	43.1477	21.9296

Table 5.5 *Scenario 1*: Impact of the perturbation strength *pStr* on the carbon footprint cost

Inst. set	Perturbation strength (*pStr*)									
	25%		50%		75%		80%		90%	
	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)
1	3.81	1.36	3.78	1.87	3.77	2.66	3.77	2.57	3.71	2.96
2	8.17	2.50	8.07	3.72	8.05	5.26	8.05	5.14	8.04	5.60
3	12.49	5.50	12.36	8.75	12.33	11.98	12.31	12.47	12.30	13.55
4	15.48	6.94	14.54	11.07	14.44	15.44	14.47	15.38	14.46	17.12
5	21.38	11.61	20.49	17.74	20.11	24.59	20.05	25.03	20.05	26.76
6	25.94	13.50	25.79	20.26	25.64	29.72	25.65	31.22	25.64	33.97
7	39.61	16.23	36.84	26.99	35.58	38.28	35.68	40.52	35.76	45.09

$pStr = 25\%$, mainly because as the strength is small, new regions in the search space cannot be reached, and the perturbation is rapidly canceled by the LS process. Also, the quality of the solution increases with higher values of *pStr* because strong perturbations increase the probability of exploring more promising solutions. However, with a too strong perturbation, as in the case of 90%, although it appears to be efficient on the tested instances, the algorithm restarts from a random point and no guarantee can be made regarding the quality of the new initial solution. Also, as the problem size increases, with a strong perturbation, the probability of starting from either bad or unfeasible solutions increases as well. Therefore, based on the results in Table 5.5, a perturbation of 75% of the components seems to be the optimal value in order to avoid greedy behavior or random restart, while allowing the best tradeoff between the quality of the solution and the CPU time.

Moreover, for evaluating the impact of the number of restarts, ILS is run with the *Swapping* method, *pStr* set to 75% and with $nbST = 25, 50, 100, 150, 200$. From Table 5.6, it can be observed that the best solutions, in terms of carbon footprint cost are obtained with high

Table 5.6 *Scenario 1*: Impact of the number of restart $nbST$ on the carbon footprint cost

Inst. set	Number of restart ($nbST$)									
	25		50		100		150		200	
	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)
1	3.77	2.66	3.70	4.89	3.66	9.60	3.64	14.62	3.64	19.67
2	8.05	5.26	8.00	9.75	7.97	18.90	7.95	28.00	7.94	37.59
3	12.33	11.98	12.28	22.50	12.26	43.69	12.24	64.18	12.24	85.09
4	14.44	15.44	14.41	27.17	14.39	52.21	14.37	76.45	14.37	100.03
5	20.11	24.59	19.98	42.20	19.96	77.77	19.95	111.61	19.94	147.23
6	25.64	29.72	25.56	54.63	25.52	106.28	25.50	158.04	25.50	210.71
7	35.58	38.28	34.95	73.56	34.78	142.61	34.75	214.86	34.61	281.65

values of $nbST$, due to the fact that a large number of restarts provides more neighborhoods to visit, which enables to increase the probability of finding better solutions. However, as $nbST$ increases, the percentage of improvement, in terms of carbon footprint cost, is significantly low compared to the increase in the CPU time. Therefore, for the remaining simulations, in order to obtain a good tradeoff between the quality of the solution and the CPU time, the *Swapping* method will be used with $pStr = 75$ and $nbST = 50$.

5.6.1.3 Perturbation Implementation - Enhanced Methods

In order to analyze the impact of complex perturbation mechanisms on the solution, the ILS algorithm is executed under $pMode_2$ and $pMode_3$, with $LIMIT = 15, 25, 35, 50$ and the *Swapping* method used as the perturbation type for $pMode_3$. The two enhanced modes are compared with the best method in $pMode_1$, the *Swapping* mechanism.

Table 5.7 shows that $pMode_2$ is the least efficient mode, as the new initial solution is based on a random selection of a perturbation type. In the case of $pMode_3$, the results show that the best solutions are obtained with $LIMIT = 35$. A low activation delay as in $pMode_{3\_15}$ will trigger the callback mechanism too frequently, while a high delay will provide less opportunity to explore new and promising configurations. Both values may therefore result in poor configurations with higher CPU time. It can also be seen that $pMode_{3\_35}$ is as efficient as the *Swapping* method, but it also has the merit to be the fastest approach. Such results confirm that the callback mechanism has an impact on the solution, and must be kept at the appropriate value in order to intensify the search in the neighborhood of the current best solution, while avoiding being trapped in local optima. Therefore, for the remaining simulations, this approach, denoted ILS_CBF, will be used as the proposed ILS method.

Table 5.7 *Scenario 1*: Impact of *pMode* on the carbon footprint cost and the CPU time

Inst. set	Perturbation modes (*pMode*)											
	$pMode_1$		$pMode_2$		$pMode_{3\_15}$		$pMode_{3\_25}$		$pMode_{3\_35}$		$pMode_{3\_50}$	
	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)	Carb (t)	CPU (s)
1	3.70	4.89	3.27	37.67	3.68	5.09	3.70	4.80	3.70	4.80	3.70	4.82
2	8.00	9.75	8.51	64.88	8.01	10.21	8.00	9.47	8.00	9.54	8.00	9.98
3	12.28	22.50	12.80	92.14	12.28	22.72	12.28	21.91	12.28	21.91	12.28	22.69
4	14.41	27.17	16.51	158.32	14.41	30.27	14.41	27.39	14.41	26.81	14.41	26.99
5	19.98	42.20	21.61	196.13	19.98	42.53	19.98	42.50	19.98	41.81	19.98	42.29
6	25.56	54.63	25.78	228.06	25.56	55.90	25.56	53.91	25.56	54.10	25.56	54.42
7	34.95	73.56	41.51	219.60	34.97	79.13	34.95	74.39	34.95	72.53	34.95	72.89

5.6.2 ILS Performance Evaluation

In order to evaluate the performance of the proposed algorithm, *ILS_CBF* is tested under different InterCloud and demand sizes, using the parameters in Table 5.2. Furthermore, the obtained results are compared with the solutions of the exact method for small-size problems, and with other baseline approaches for larger instances.

5.6.2.1 Comparison with the Exact Method

The efficiency of the proposed method is first evaluated by comparing the obtained results with a lower bound that is defined as the exact solution (*EXACT_CBF*) and calculated using a linear program and the CPLEX solver. The results are presented in Table 5.8, where column 2 gives the problem size in terms of number of data centers (D), chassis (X), servers (S) and VMs (V); columns 3 and 4 show the optimal carbon footprint cost and the associated CPU time respectively; and the last columns give the average distance (in %) of the solutions obtained with *ILS_CBF* from the lower bounds evaluated (the carbon footprint being identical over 10 runs), and the related computing time (minimal, mean and maximal).

Table 5.8 shows that the carbon footprint costs obtained from *ILS_CBF* are very close to the lower bounds, ranging from 0% to a maximum distance less than 2.6%. The "*" sign, in instance #11, indicates that the instance cannot be solved to optimality due to resource limitation. Computational results also indicates that the CPU time for the exact method follows an exponential growth, while the computing time for the proposed approach approximately increases linearly, without any combinatorial explosion. With an average computing time less than 3 seconds, a maximum CPU time (among all the tested instances) of about 3.9 seconds and carbon footprint costs close to the lower bounds, the proposed approach *ILS_CBF*

Table 5.8 *Scenario 1*: Carbon footprint and CPU time comparison between *EXACT_CBF* and *ILS_CBF*

Inst. set	Problem size	*EXACT_CBF*		*ILS_CBF*			
		Carb	CPU	Dist. to LB	CPU (s)		
	(D,X,S,V)	(Kg)	(s)	(%)	Min	Mean	Max
1	1, 4, 20, 20	1389.76	8.23	1.51	0.54	0.60	0.82
2	1, 4, 40, 40	1496.15	14.91	0.00	0.55	0.58	0.65
3	1, 4, 40, 80	1818.55	36.61	0.00	0.58	0.60	0.67
4	1, 4, 60, 60	1707.41	96.77	0.00	0.60	0.62	0.65
5	1, 4, 60, 120	2131.16	186.25	0.00	0.65	0.68	0.69
6	1, 8, 40, 40	1496.15	14.97	0.00	0.54	0.59	0.63
7	1, 8, 40, 80	1818.55	21.11	0.00	0.56	0.58	0.63
8	1, 8, 80, 160	2451.32	187.06	0.00	0.71	0.75	0.81
9	1, 8, 120, 120	2131.16	285.17	0.00	0.73	0.77	0.85
10	1, 8, 120, 240	2702.15	977.63	0.00	1.05	1.11	1.15
11	1, 12, 180, 360	*	26712.42	*	1.73	1.85	1.96
12	2, 8, 40, 20	1031.93	56.08	0.00	0.55	0.56	0.59
13	2, 8, 80, 40	1081.71	85.44	0.00	0.56	0.59	0.63
14	2, 8, 80, 80	1295.86	304.03	0.00	0.62	0.64	0.71
15	2, 8, 120, 60	1225.59	280.80	0.00	0.59	0.63	0.68
16	2, 8, 120, 120	1504.33	1055.33	0.00	0.74	0.78	0.84
17	2, 16, 80, 40	1081.71	83.98	0.00	0.55	0.58	0.63
18	2, 16, 80, 80	1295.86	230.13	0.00	0.61	0.63	0.69
19	2, 16, 160, 80	1295.86	423.69	0.00	0.67	0.71	0.76
20	2, 16, 160, 160	1712.38	2874.28	0.00	0.95	0.98	1.03
21	2, 16, 240, 120	1504.33	1588.47	0.00	0.90	0.95	1.00
22	2, 16, 240, 240	1940.43	24711.67	0.00	1.49	1.58	1.68
23	2, 24, 120, 60	191.98	189.39	0.00	0.60	0.63	0.68
24	2, 24, 240, 120	238.14	1420.03	0.00	0.87	0.94	1.01
25	2, 24, 240, 240	301.40	7677.20	2.60	1.38	1.45	1.55
26	2, 24, 360 180	268.44	6439.69	0.00	1.37	1.41	1.46
27	2, 24, 360, 360	399.10	44558.67	1.92	2.75	2.85	2.91
28	3, 12, 60, 20	855.93	502.09	0.00	0.54	0.56	0.59
29	3, 12, 60, 40	971.02	695.06	0.00	0.55	0.58	0.60
30	3, 12, 120, 40	971.02	1070.88	0.00	0.58	0.62	0.66

clearly outperforms the exact method *EXACT_CBF*, allowing an excellent tradeoff between the quality of the solution and the CPU time.

5.6.2.2 Comparison with other approaches

In order to analyze the general performance of the *ILS_CBF* method and assess its efficiency, the proposed algorithm is executed under different problem sizes that cannot be solved to optimality and the results are compared with those obtained from the adaptation of well-known algorithms: an instinctive approach, $Green\_Data\_Center\_First_{Improved}$ ($GDCF_I$) that sorts the data centers based of their greenness factor and uses a modified *Best Fit* algorithm (Mishra et Sahoo, 2011); an adapted version of the *First_Fit_Decreasing* method (FFD_A)

Table 5.9 *Scenario 1*: Carbon footprint and CPU time comparison between *ILS_CBF* and other reference methods

Inst. set	Problem size	*ILS_CBF*		*GDCF_I*		*FFD_A*		*BFD_A*	
		Carb	CPU	Carb	CPU	Carb	CPU	Carb	CPU
	(D,X,S,V)	(t)	(s)	(t)	(s)	(t)	(s)	(t)	(s)
1	1, 12, 60, 60	1.606	0.636	1.626	0.004	1.639	0.001	1.626	0.003
2	1, 12, 120, 120	1.982	0.760	2.031	0.015	2.010	0.002	2.031	0.012
3	1, 12, 120, 240	2.551	1.037	2.659	0.026	2.570	0.003	2.659	0.024
4	1,12, 180, 180	2.270	1.030	2.346	0.031	2.293	0.004	2.346	0.026
5	1, 12, 180, 360	3.300	1.847	3.506	0.054	3.311	0.006	3.506	0.052
6	3, 12, 60, 40	0.995	0.649	0.995	0.001	1.115	0.001	0.995	0.003
7	3, 24, 120, 80	1.201	0.746	1.201	0.004	1.296	0.002	1.201	0.009
8	3, 30, 300, 1800	25.563	54.802	28.029	0.157	27.413	0.064	28.029	0.339
9	3, 30, 300, 2100	34.947	72.390	46.688	0.194	44.637	0.084	46.688	0.377
10	4, 40, 200, 1000	6.286	6.343	6.286	0.047	6.286	0.019	6.286	0.162
11	4, 40, 400, 2000	8.547	38.446	8.547	0.170	8.547	0.064	8.547	0.621
12	6 ,60, 600, 3000	14.494	116.454	14.494	0.266	14.494	0.149	14.494	1.391
13	6, 120, 600, 3000	18.289	88.277	18.289	0.269	18.289	0.154	18.289	1.414
14	8, 160, 800, 4000	22.809	207.743	23.012	0.410	22.864	0.248	23.012	2.500
15	10, 100, 500, 2500	14.141	54.204	14.388	0.161	14.181	0.099	14.388	0.997

(Wood *et al.*, 2009) and a modified version of the *Best_Fit_Decreasing* algorithm (BFD_A) (Beloglazov et Buyya, 2010).

From the results in Table 5.9, it can be seen that *ILS_CBF* always provides the best carbon footprint cost. The reason is that the proposed heuristic tends to refine the initial solution by visiting the neighborhood of the current best configuration. Regarding the greedy algorithms, although extremely fast, they usually end up in poor configurations mainly because they stop once the VMs are sorted and assigned, without any further exploration. The carbon footprint cost difference is also due to the fact that those greedy algorithms use an identical temperature value for all the data centers, while the optimal solution may lead to different values of data centers' temperature. Moreover, one interesting observation is that while in a relatively small search space (as in instances 1 to 5), the baseline methods cannot be as efficient as the proposed heuristic, they provide the same solutions as the *ILS_CBF* algorithm for larger problem sizes (as in instances 11 and 12). Due to the heterogeneous nature of the InterCloud environment, no guarantee can be made regarding the performance of the greedy algorithms which can perform VM placement with an error margin of up to 33.6%, as in instance 9. It becomes therefore more suitable to use the proposed *ILS_CBF*, as the algorithm has higher flexibility to explore the search space and provides good tradeoffs between the quality of the solution and the computational time.

5.7 Conclusion

In this paper, we proposed a new mathematical model along with an implementation of the ILS metaheuristic in order to perform an efficient VM placement, with the view of minimizing the carbon footprint of an InterCloud environment. Simulations were conducted in order to evaluate the performance of the proposed model. In the first tests, the impact of several key parameters was analyzed in order to identify the best values that are able to reduce the carbon footprint costs. In particular, the best configurations were obtained with $pMode_3$, $pType = 2$, $pStr = 75$, $nbST = 50$ and $LIMIT = 35$. Extended simulations were run in order to assess the efficiency of the proposed ILS methods. Comparison with the exact method *EXACT_CBF* shows that the computational time of *ILS_CBF* increases linearly in the length of the input while the carbon footprint cost obtained is usually very close to the lower bounds, with a maximum distance of about 2.6%, providing therefore good tradeoffs between the quality of the solutions and the computational time. For large instances of the problem, results based on the comparison with well-known greedy algorithms demonstrate that the proposed resolution approach may allow carbon footprint savings of up to 33.6%.

At this point, several research avenues are possible: the network power consumption could be included in the mathematical model, as different types of applications, spanning multiple VMs and leading to a non-negligible inter-VM traffic, can be hosted in an InterCloud. Furthermore, the model could be extended by considering efficient techniques, such as free cooling or data center heat redistribution, in order to increase the cooling system efficiency. Migration techniques could also be integrated with the purpose of adapting the model to online problems. Finally, further works could be oriented towards combining the proposed ILS algorithm with exact methods in order to optimally solve the VM placement problem in polynomial computational time.

Appendix A Parameters

The temperature and maximal power of a chassis at temperature τ are given by (Pakbaznia et Pedram, 2009):

$$\tau_x^{out}(\tau) = P_x^{CH}(\tau) * \left(R_x^0/\tau\right) + \tau, \quad \forall x \in X \tag{A.1}$$

$$P_x^M(\tau) = \left(\tau_x^{out\_M} - \tau\right) / \left(R_x^0/\tau\right), \quad \forall x \in X \tag{A.2}$$

The outlet temperature difference of a chassis x is evaluated as follows:

$$\begin{aligned} T_x^{\Delta} &= n_x^F * R_x^0 * \eta_x^0 * \left| t_2^2 - t_1^2 \right| \\ &+ |t_2 - t_1| * \left(1 - \frac{R_x^0 * \left(p_x^X + \sum_{s \in S} \alpha_{sx}^X * z_s^S * P_s^S \right)}{t_2 * t_1} \right), \quad \forall x \in X \end{aligned} \tag{A.3}$$

Equations (A.4) and (A.5) determine the number of VMs hosted on a server and the number of active servers of a chassis:

$$n_s^S = \sum_{v \in V} z_{vs}^V, \quad \forall s \in S \tag{A.4}$$

$$n_x^X = \sum_{s \in S} z_s^S * \alpha_{sx}^X, \quad \forall x \in S \tag{A.5}$$

CHAPITRE 6 ARTICLE 3 : A HYBRID APPROACH FOR OPTIMIZING CARBON FOOTPRINT IN INTERCLOUD ENVIRONMENT

Auteurs : Valérie Danielle Justafort, Ronald Beaubrun et Samuel Pierre.
Revue : Soumis dans le journal *IEEE Transactions on Services Computing*, en Octobre 2015.

Abstract

This paper focuses on the problem of workload placement in an InterCloud with the view of minimizing the carbon footprint of such a computing environment. In order to reduce the ecological impact of the data center Greenhouse Gas (GhG) emissions, this paper addresses the problem as a whole, by proposing a global mathematical formulation, based on the joint optimization of the Virtual Machine (VM) placement and their related traffics, along with a workload consolidation method and a cooling maximization technique that considers the dynamic behavior of the cooling fans. As the Virtual Machine Placement Problem (VMPP) is classified as an NP-hard problem, with the addition of the traffic embedding, the problem becomes more complex and stays NP-hard. Therefore, we propose a hybrid approach, for solving such problem and find good feasible solutions in a polynomial time. The results obtained from comparing with the exact method and other reference approaches help in assessing the efficiency of the proposed algorithm, as the carbon footprint costs are relatively close to the lower bound, with an average gap of about 3%, and found within a reasonable amount of time.

6.1 Introduction

Cloud Computing has recently emerged as the new trend in the Information and Communication Technology (ICT) sector, allowing end users to run their applications, encapsulated in Virtual Machines (VMs), in a secure computing environment (Chowdhury *et al.*, 2009), while ensuring proper performance isolation among co-located VMs (Sharifi *et al.*, 2012). However, security or redundancy issues (Bonde, 2010), which may impact the Virtual Machine (VM) placement, have been ignored. Also, due to the rapid growth in the number and use of large data centers and the increasing energy cost (Greenpeace, 2010), (Mazzucco *et al.*, 2010), various workload-aware consolidation techniques have been proposed in order to tackle the servers' power minimization problem (Von Laszewski *et al.*, 2009), (Fang *et al.*,

2013a), under defined performance constraints. Approaches for improving cooling cost have also been developed in order to further reduce data centers' power consumption (Pakbaznia et Pedram, 2009). However, the dynamic behavior of IT equipment fans which influences the cooling efficiency at high temperatures (Moss et Bean, 2009) has often been neglected.

Furthermore, clouds' high energy consumption does not only affect providers' profits, but highly impacts climate change, as data centers' carbon footprint represents 2% of the total Greenhouse Gas (GhG) emissions (Moghaddam *et al.*, 2012), and is increasing. Although power minimization techniques have been considered in order to reduce single clouds' ecological impact, they cannot be applied in InterCloud environment, where energy consumption does not always reflect the carbon footprint (Moghaddam *et al.*, 2011). In this context, minimizing the GhG emissions of an interCloud has lately attracted a great deal of attention, as intelligent methods need to be implemented to properly assign the VMs and improve the cooling efficiency, in order to jointly consider data centers' greenness factor and their power consumption.

The aforesaid challenges have been tackled in our previous works (Justafort *et al.*, 2014), (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a), where carbon footprint optimization has been performed in an InterCloud setup while considering a workload defined by standalone VMs. However, with the rising number of applications spanning multiple VMs and making use of the switches and links to route the inter-VMs traffic demands, the network power consumption and their associated carbon footprint can no longer be neglected. As the latter accounts for about 25% of the equipment power consumption (Fang *et al.*, 2013a), efforts have been deployed in order to minimize the network consumption. However, optimizing the VM placement or the traffic routing alone is too restrictive, as VM consolidation may introduce network congestion (Zheng *et al.*, 2014), and traffic routing may result in servers' resource wastage (Jiang *et al.*, 2012). Having a control on both entities will help efficiently optimize data centers' resources and reduce IT total power consumption. Therefore, in the context of a more general workload placement, which includes complex applications involving inter-VMs traffic demands, reducing the carbon footprint of an InterCloud environment becomes even more challenging as it remains unclear how to jointly consider techniques such as data center selection, smart VM consolidation and cooling efficiency optimization methods, introduced in (Justafort *et al.*, 2014), (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a), with intelligent traffic demand routing approaches, which select the best power-efficient switches while preventing network congestion.

In this paper, we are therefore interested in a global approach to the Virtual Machine Placement Problem (VMPP). Also, as many aspects of the VMPP refer to well-known NP-hard

problems, such as the Knapsack or Bin-Packing Problem (BPP) (Wu, 2013), exact algorithms are unsuitable to solve moderate and large instances of the problem, as in common optimization solvers, the computing time is exponential, due to the numerous integer variables of the models. Approximate methods are therefore highly required in order to obtain sub-optimal solutions in a polynomial time.

In order to address the challenges stated above, this paper proposes an InterCloud planning model, named $CB_G\_OPT\_SLA$, to reduce the total carbon footprint of such a computing environment, and an heuristic method, ITS_G_{CBF}, to solve large instances of the problem. The main contributions of this paper are listed below:

— We address the problem related to the ecological impact of InterCloud environment and improve the model presented in (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a), by considering traffic routing optimization techniques in order to efficiently utilize the data centers' resources and jointly minimize servers', network resources and cooling carbon footprint;

— Using modeling techniques for Integer Programming (IP) models (Williams, 2013), the general workload placement problem has been stated as a Mixed-Integer Non-linear Programming (MINLP) problem in order to minimize the carbon footprint in InterCloud environment;

— Although optimization solvers provide the optimal value, they can only scale up to topologies with small size. Therefore, an Iterated Tabu Search (ITS) heuristic with acceptable time complexity has been developed in order to obtain good feasible solutions for large instances of the problem. ITS has been chosen as it combines the simplicity and effective applicability of the Iterated Local Search (ILS) heuristic (Congram *et al.*, 2002), along with the memory mechanisms implemented in the Tabu Search (TS) algorithm, which is used as the local search process (Bilal *et al.*, 2014).

The rest of the paper is organized as follows. Section 6.2 discusses relevant works related to the problem. The system model is presented in Section 6.3, whereas the problem formulation is described in Section 6.4. Section 6.5 presents the adaptation of the proposed algorithm, ITS_G_{CBF}. Experiments and simulations are illustrated in Section 6.6, whereas conclusion and future works are presented in Section 6.7.

6.2 Related work

Over the past few years, as power consumption cost has largely influenced the operating cost of data centers, besides hardware power management techniques (Li, 2015), consolidation methods have been used to reduce the number of active hosts (Buyya *et al.*, 2010),

(Dong *et al.*, 2013) and (Liu *et al.*, 2015), or to minimize their power consumption (Dupont *et al.*, 2012), (Larumbe et Sanso, 2012) and (Shuja *et al.*, 2014). Techniques using energy-performance optimal points (Srikantaiah *et al.*, 2008) or a consolidation fitness metric, the Consolidation Fitness (CF) coefficient (Sharifi *et al.*, 2012), or an interference model (Kim *et al.*, 2013), have been considered in order to cope with Service Level Agreement (SLA) violations. However, in the VM placement process, not only the intrinsic performance of the co-allocated VMs should be considered, but security or redundancy constraints should also be integrated (Bonde, 2010), as they can alter the Quality of Service (QoS) expressed in the SLA. Also, as the Computer Room Air Conditioner (CRAC) overhead accounts for about 30% of the total power consumption of a data center, VM placement approaches based on heat extracting methods (Patel *et al.*, 2003), (Sharma *et al.*, 2005) or using the heat flow model presented in (Tang *et al.*, 2006), have been used to optimize the energy cost under temperature constraints (Polverini *et al.*, 2014) or to minimize the server and cooling power cost of a data center (Pakbaznia et Pedram, 2009). However, the dynamic behavior of the IT fans studied in (Moss et Bean, 2009) has not been considered. Lee presented a case study of the impact of data center inlet temperature on energy efficiency (Lee, 2012), but ignored the dimension related to the VM placement process.

Recent works have addressed the carbon footprint minimization problem (Liu *et al.*, 2012), (Goiri *et al.*, 2015). However, the carbon footprint problem was tackled considering a single-data center setup and no mechanism aiming at optimizing the cooling efficiency was considered when only brown energy sources are available. Other researches have focused on the ecological impact of an InterCloud (Moghaddam *et al.*, 2012), (Larumbe et Sanso, 2012) and (Van Heddeghem *et al.*, 2012). For instance, Liu *et al.* (2015) also explored the benefits of geographical load balancing with regards to carbon emissions and in (Liu *et al.*, 2011), they proposed an algorithm to find the optimal percentage of wind/solar energies in order to reduce the brown energy consumption. Doyle *et al.* (2013) developed a method based on Voronoi partitions in order to simultaneously reduce the delay, the energy and the carbon footprint cost while Khosravi *et al.* (2013) addressed the problem of increase in carbon footprint of an InterCloud by considering the greenness factors of the cloud sites, their efficiency or Power Usage Effectiveness (PUE) and the servers' power profiles. Although these works tried to tackle the carbon footprint minimization problem, they failed at considering the cooling cost and the dynamic behavior of the server fan, which largely impact a data center power consumption and the equivalent carbon footprint.

Regarding our prior works, Justafort *et al.* (2014) and Justafort *et al.* (2015b) presented a model that eliminates the short-sighted aspect of the VM planning: (1) they performed smart VM consolidation while considering performance degradation as well as co-locations

constraints, (2) they explored the power reduction of the CRAC against the potential increase in IT power at high temperatures, (3) they extended the problem to consider the InterCloud carbon footprint cost and (4) they formulated the problem as a MINLP in order to find the optimal solution using an AMPL/CPLEX program. However, regarding the workload, only the VM placement has been addressed. However, with the emergence of applications spanning multiple VMs and traffic demands consuming nearly 25% of the data center dissipation (Fang *et al.*, 2013a), reducing cloud network power consumption has attracted significant attention. Some works have therefore focused on consolidating the inter-VMs traffic flows onto a few network elements (Heller *et al.*, 2010), Shang *et al.* (2010) or developing routing techniques in order to minimize their power consumption (Fang *et al.*, 2013b) and (Larumbe et Sanso, 2012). However, as the VM placement and the flow routing processes are two mutually-dependent problems, they cannot be tackled separately. In this context, recent works have started to focus on the two aspects: techniques based on a joint optimization of the VM placement and the traffic routing have been introduced, however with the purpose of minimizing the network congestion in the long run (Jiang *et al.*, 2012), the traffic communication cost (Meng *et al.*, 2010), the server power and the network delay (Fang *et al.*, 2013a), or the network power consumption (Fang *et al.*, 2013b). The servers and network power consumption have been simultaneously considered in (Zheng *et al.*, 2014) and (Larumbe et Sanso, 2013), where the problem of an InterCloud power minimization has also been analyzed. However, in this latter work, virtualization techniques has been ignored as the VM placement was restricted to placing one component per physical host.

Also, as the workload placement problem is NP-hard, even in the offline version, numerous works attempted to propose resolution methods in order to solve large instances of the problem: First Fit, Best Fit and Worst Fit algorithms have been introduced in (Fang *et al.*, 2013a), while an adaptation of the Simulated Annealing heuristic has been proposed in (Wu, 2013). In (Feller *et al.*, 2011), an algorithm based on the Ant Colony Optimization metaheuristic has been designed to compute the VM placement, an ILS approach has been used for request partitioning in (Leivadeas *et al.*, 2013) and a TS adaptation has been proposed in (Larumbe et Sanso, 2013) to assign applications to data centers. Justafort *et al.* also proposed, in (Justafort *et al.*, 2015a), an adaptation of the ILS method in order to solve large instances of the carbon footprint minimization problem presented in the same work.

The present paper complements the work in (Justafort *et al.*, 2014), (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a) and includes the following contributions: (1) a new workload model composed of a mixture of stand-alone VMs as well as applications spanning multiple components, with their traffic demands and their associated location constraints; (2) an evaluation of the network power consumption and the equivalent carbon footprint cost; (3)

a joint optimization of the VM placement and the traffic routing for minimizing the carbon footprint of an InterCloud environment and (4) a hybrid approach, ITS_G_{CBF}, based on metaheuristic techniques, for solving large instances of the VMPP. An important characteristic of our algorithm is the use of two levels of local search mechanisms that help avoiding local optima and cycles, along with a perturbation method that intensifies the search around the current optimum method and a diversification process that performs long jumps in the search space, in order to explore unvisited regions.

6.3 System environment

Two entities define the proposed framework: the InterCloud environment and the workload representation.

6.3.1 InterCloud environment

The InterCloud is made of a VM Management System to perform workload placement, a set of data centers and the clients with their applications. A data center is composed of racks of chassis, with multiple servers sharing the same fans and power supply. The servers, interconnected in a tree topology, as shown in Fig. 6.1 (Meng *et al.*, 2010), are heterogeneous, provide resources in terms of CPU cycles, memory size and disk, and can run multiple VMs. The chassis level is considered as the temperature spacial granularity (Pakbaznia et Pedram, 2009), and its inlet temperature depends on the air supplied by the CRAC system, due to physical separators (Posladek, 2008).

6.3.2 Client and workload representation

The applications, submitted by the clients, are encapsulated in VMs, characterized by a multidimensional resource vector consisting of the following components: [cpu, $disk$, $memory$]. The workload can be of two types: stand-alone VMs or applications spanning multiple VMs and with inter-VMs traffic demands. Due to technical constraints, for redundancy or security purpose (Bonde, 2010), interference constraints require that certain VMs are not co-located and affinity coefficients are derived accordingly. Clients may also impose location constraints, in terms of potential data centers to host the applications. Moreover, the VM placement constraints related to performance degradation, which was introduced in (Sharifi *et al.*, 2012) and adapted in (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a), are also considered in this paper.

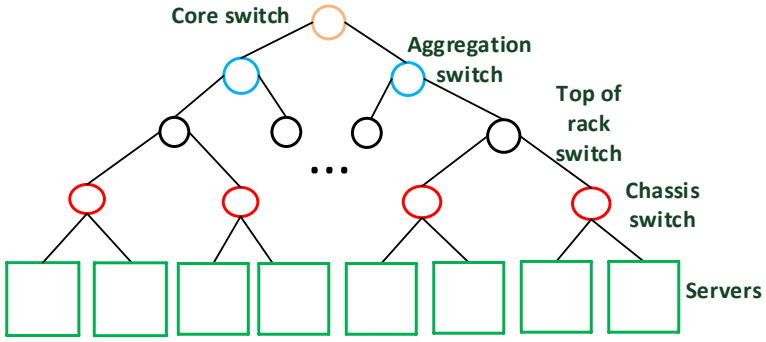


Figure 6.1 The tree topology

6.4 Problem statement

This section presents an overview of the workload placement along with the mathematical formulation of the optimization problem.

6.4.1 VM placement overview

In order to perform carbon footprint optimization in an InterCloud environment, the workload placement problem should be considered as a whole. Due to the heterogeneity of the physical infrastructure, VMs should be placed on the most power-efficient servers, while performing smart consolidation and considering the green factor of the data centers. Also, the optimal indoor temperature of each data center should be determined in order to cope with the complex interactions between the fans' power consumption and the cooling dissipation. Furthermore, as the applications may span multiple VMs, inter-VMs traffic can no longer be neglected. Therefore, an efficient VM and traffic embedding, considering the server and network resource characteristics along with the proximity constraints of the communicating VMs, is therefore a must.

6.4.2 Assumptions

In addition to the assumptions made in (Justafort *et al.*, 2014), (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a), we refer to a flow as a traffic demand between a pair of VMs, and a path as a set of interconnected links between 2 servers of a data center. Moreover, in order to avoid delay violation and inter-data centers traffic, the components of one application will be placed in a unique data center. Moreover, we assume that the affinity coefficients are

derived, following the clients submission, but prior to the VM placement process. Also, VM migration is not being addressed.

6.4.3 Model formulation

Using IP techniques, the global model for the carbon footprint minimization of an InterCloud environment is presented in the following.

6.4.3.1 Notation

Different sets, parameters and variables are used to describe the optimization problem. In order to avoid repetition, part of the notation and some constraints introduced in our previous works, (Justafort *et al.*, 2014), (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a), will not be presented here.

Sets: Let D be the set of data centers and E the set of energy sources. X is the set of chassis, S, the set of servers and K, the set of available resources (CPU, memory and disk). Let T, L and P be respectively the set of switches, links and paths. The set of clients, VMs and flows are denoted by C, V and F. Let A be the set of applications, and V_a^A and D_a^A be respectively the set of VMs and the potential data centers for application $a \in A$.

Power consumption parameters: Let p_x^X and η_x^0 be respectively the idle power and the fan power coefficient of chassis $x \in X$. p_s^S is the idle power of server $s \in S$ and a_{sk}, is the energy consumption under 100% utilization of the resource $k \in K$. Let p_t^T be the idle power of switch $t \in T$ and a_t be its power consumption 100% utilization.

Other parameters: Let V_{vk_1}, V_{vk_2}, V_{vk_3} be the CPU, disk and memory requirements of VM $v \in V$, and b_f, the bandwidth requirement of flow $f \in F$. C_{sk_1}, C_{sk_2}, C_{sk_3} are the CPU, disk and memory capacity of server $s \in S$. Let C_t^T and C_l^L be respectively the bandwidth capacity of switch $t \in T$ and link $l \in L$. Let Π_f^O and Π_f^D be the origin and destination VMs of flow $f \in F$, Θ_p^O and Θ_p^D, be the origin and destination servers of path $p \in P$. n_x^F is the number of fans of chassis $x \in X$. Let G_e be the carbon footprint, in Kg per CO_2, of energy source $e \in E$; e_{ed}, the percentage of power due to energy source $e \in E$ in data center $d \in D$ and G_d^{TOT}, the greenness factor of data center $d \in D$.

Decision Variables: Let z_t^T be a 0-1 variable such as $z_t^T = 1$ if switch $t \in T$ is active, z_x^X be

a 0-1 variable such as $z_x^X = 1$ if chassis $x \in X$ is active and z_s^S a 0-1 variable such as $z_s^S = 1$ if server $s \in S$ is active. z_{ad}^A is a 0-1 variable, such as $z_{ad}^A = 1$, if application $a \in A$ is located in data center $d \in D$, z_{vs}^V is a 0-1 variable such as $z_{vs}^V = 1$ if VM $v \in V$ is hosted on server $s \in S$ and z_{fp}^P is a 0-1 variable, such as $z_{fp}^P = 1$ if flow $f \in F$ is routed through path $p \in P$.

Other variables: Let $u_{sk_1}, u_{sk_2}, u_{sk_3}$ be respectively the CPU, disk and memory usage of server $s \in S$. u_t^T and u_l^L are the bandwidth usage of switch $s \in S$ and link $l \in L$, and τ_d is the supplied air temperature in data center $d \in D$.

Other optimization parameters: Let α_{sx}^X be a 0-1 parameter, such as $\alpha_{sx}^X = 1$ if server $s \in S$ belongs to chassis $x \in X$, and β_{xd} is a 0-1 parameter where $\beta_{xd} = 1$ if chassis $x \in X$ belongs to data center $d \in D$. δ_{tp}^T and κ_{td}^T are 0-1 parameters, such as $\delta_{tp}^T = 1$ if switch $t \in T$ belongs to path $p \in P$ and $\kappa_{tp}^T = 1$ if switch $t \in T$ is located in data center $d \in D$. The same reasoning applies to δ_{lp}^L for link $l \in L$.

6.4.3.2 Power models

According to Pelley *et al.* (2009), the cooling system, the computing nodes and the switches are the main contributors of the electricity bill of large data centers. As communicating VMs should be placed in the same data center, in order to ensuring minimum delay, backbone routers and links power consumption are neglected.

Fan power

The IT fans considered in this paper continuously vary their power over a range of inlet temperatures (Moss et Bean, 2009). Considering the fan law in (Lee, 2012), the fan power consumption $P_x^F(\tau)$ at temperature τ is given by:

$$P_x^F(\tau) = \eta_x^0 * \tau^3, \quad \forall x \in X \tag{6.1}$$

Server power

The server power model suggested by Sharifi *et al.* (2012) and Lee et Zomaya (2012) is used in this paper and is obtained as follows:

$$P_s^S = p_s^S + a_{sk_1} * u_{sk_1} + a_{sk_2} * u_{sk_2}, \quad \forall s \in S \tag{6.2}$$

with the processor usage of a server, for instance, given by:

$$u_{sk_1} = \sum_{v \in V} \left(z_{vs}^{V} * V_{vk_1} \right) / C_{sk_1}, \quad \forall s \in S \tag{6.3}$$

Chassis power consumption

The overall chassis power consumption $P_x^{CH}(\tau)$ includes its base power, along with the active servers and the associated fans power dissipation:

$$P_x^{CH}(\tau) = p_x^X + n_x^F * P_x^F(\tau) + \sum_{s \in S} \alpha_{sx}^X * z_s^S * P_s^S, \quad \forall x \in X \tag{6.4}$$

Switch power

As for the server, the switch power consumption is based on its idle power and its usage (see (6.6)) as follows:

$$P_t^T = p_t^T + a_t * u_t^T, \quad \forall t \in T \tag{6.5}$$

with the bandwidth usage of switch t obtained as follows:

$$u_t^T = \sum_{p \in P} \sum_{f \in F} \delta_{tp}^T * z_{fp}^P * b_f / C_t^T \quad \forall t \in T \tag{6.6}$$

Cooling system power

The cooling system power consumption of a data center d depends on its efficiency, described by its Coefficient of Performance (COP), and is given by:

$$P_d^{CO}(\tau_d) = P_d^{IT}(\tau_d) / \mathrm{COP}_d(\tau_d), \quad \forall d \in D \tag{6.7}$$

with the COP curve quantified in terms of the supplied temperature (Pakbaznia et Pedram, 2009).

6.4.3.3 The cost function

The carbon footprint minimization depends on the data centers power consumption, namely the IT equipment power and the CRAC power dissipation, and their greenness factor. The IT equipment power consumption of a data center d $P_d^{IT}(\tau_d)$ is given by:

$$P_d^{IT}(\tau_d) = \sum_{x \in X} \beta_{xd} * z_x^X * P_x^{CH}(\tau_d) + \sum_{t \in T} \kappa_{td}^T * z_t^T * P_t^T, \quad \forall d \in D \tag{6.8}$$

From (6.7), the total power consumption inside a data center $P_d(\tau_d)$ is given by:

$$P_d(\tau_d) = P_d^{IT}(\tau_d) * [1 + 1/\mathrm{COP}(\tau_d)], \quad \forall d \in D \tag{6.9}$$

where the PUE of a data center $\rho_d(\tau_d)$ is given by:

$$\rho_d(\tau_d) = 1 + 1/\mathrm{COP}(\tau_d), \quad \forall d \in D \tag{6.10}$$

The power-carbon conversion rate is given by (Moghaddam *et al.*, 2012):

$$G_d^{TOT} = \sum_{e \in E} G_e * e_{ed}, \quad \forall d \in D \tag{6.11}$$

6.4.3.4 The model

The carbon footprint minimization of the InterCloud environment is expressed as follows:

$$MIN \quad \sum_{d \in D} G_d^{TOT} * \rho_d(\tau_d) * P_d^{IT}(\tau_d) \tag{6.12}$$

In addition to the constraints presented in (Justafort *et al.*, 2014), (Justafort *et al.*, 2015b) and (Justafort *et al.*, 2015a), the problem is subjected to the following constraints:

Each flow should be routed through a single path

$$\sum_{p \in P} z_{fp}^P = 1, \quad \forall f \in F \tag{6.13}$$

Equations (6.14), (6.15) and (6.16) ensure that all the components of an application are placed in a unique data center.

$$\sum_{d \in D} z_{ad}^A = 1, \quad \forall a \in A \tag{6.14}$$

$$\sum_{d \in D_a^A} z_{ad}^A = 1, \quad \forall a \in A \tag{6.15}$$

$$\left\| V_a^A \right\| * z_{ad}^A \leq \sum_{v \in V_a^A} \sum_{s \in S} z_{vs}^V * \alpha_{sd}^D, \quad \forall a \in A, d \in D \tag{6.16}$$

Equations (6.17) and (6.18) relate the flow routing to the origin and destination of the traffic demands.

$$z_{fp}^P \leq z_{\Pi_f^O, \Theta_p^O}^V, \quad \forall f \in F, p \in P \tag{6.17}$$

$$z_{fp}^{P} \leq z_{\Pi_f^D, \Theta_p^D}^{V}, \quad \forall f \in F, p \in P \tag{6.18}$$

Equation (6.19) expresses the switch activation constraints.

$$z_t^T \geq u_t^T, \quad \forall t \in T \tag{6.19}$$

Constraints (6.20) and (6.21) state that the bandwidth usage of a switch or a link should not exceed 1.

$$u_t^T \leq 1, \quad \forall t \in T \tag{6.20}$$

$$u_l^L \leq 1, \quad \forall l \in L \tag{6.21}$$

with the bandwidth usage of link l given by:

$$u_l^L = \sum_{p \in P} \sum_{f \in F} \delta_{lp}^L * z_{fp}^P * b_f / C_l^L \quad \forall l \in L \tag{6.22}$$

Decision variables can only be Boolean while the non-negativity constraints reduce the domain of the other variables to integers or positive real numbers.

6.5 The proposed $\boldsymbol{ITS\_G_{CBF}}$

In order to solve large instances of the VMPP, we propose an hybrid algorithm, namely ITS_G_{CBF}, combining two meta-heuristics: ILS and TS.

6.5.1 Basic principles

An ILS algorithm begins with a solution and explores the search space in order to determine the best configuration. Each iteration starts with a solution generated with a perturbation operator, trying to obtain a new optimum using a Local Search (LS) operator. A given acceptance criterion determines whether the optimum should be kept or not.

The TS heuristic uses a Tabu list which stores the recent moves, so they cannot be performed for T_l iterations, avoiding therefore cycles and local optima. However, the tabu status of a move is overridden if the resulting solution is better than the current best one. The exploration stops if all the moves are tabu, or when the actual best solution has not been improved for K_m iterations. Enhanced memory mechanisms are often used to solve harder problems. To achieve better results, we propose an hybrid algorithm where TS is used as the LS operator of the ILS heuristic.

6.5.2 Adaptation of ITS_G_{CBF}

The proposed algorithm is defined by a two-level LS processes, namely TS_1 and TS_2, combined with two perturbation phases and a long-term memory mechanism. ITS_G_{CBF} starts with an initial solution followed by an execution of TS_1 and TS_2 (step 1). Then in each iteration of the first perturbation phase, a run of TS_1 follows the perturbation of the best solution. If the latter has not been improved after K_p iterations, the second perturbation phase is triggered in order to intensify the search in the neighborhood of the best solution. This phase consists of iteratively executing TS_2 using the best solution perturbed as a new beginning, and ends once no improvement is observed. The Diversification process is therefore invoked in order to generate a new starting point and ITS_G_{CBF} returns to step 1. The acceptance criterion only allows improving solutions and ITS_G_{CBF} stops after N_s calls of the Diversification process. In order to adapt the algorithm, the solution space, the initial solution, the perturbation and the LS operators, along with the memory mechanism need to be fine tuned.

6.5.2.1 Solution Space and Initial Solution

A solution of the ITS heuristic is determined by the variables z_{vs}^V and τ_d satisfying the problem constraints, besides the ones related to resource capacity, chassis power or temperature limitations and SLA requirements. With these values set, the other variables can be computed using the equations in Section 6.4. As a solution s can be unfeasible, due to constraints violations, a penalty cost, $P(s)$, is also considered in the evaluation cost, $E(s)$, besides the intrinsic cost, $C(s)$, calculated with (6.12).

In order to generate the initial solution, the applications are first sorted in a ascending order of their number of locations, then the VMs are listed in a descending order of their resource requirements whereas the servers are sorted such as the ones in the cleanest data centers are on top of the list. Then, for each temperature value, the applications are first located, followed by a run of the First Fit algorithm on the remaining VMs. An application is placed by assigning each VM to the first suitable server, while considering the location constraints and ensuring that all the components are in the same data center. Once the VMs are hosted, the traffics are routed through the network and the configuration with the lowest cost is chosen.

6.5.2.2 LS Mechanisms: TS_1 and TS_2

The LS process consists of moving, iteratively, from an actual configuration to a neighbor solution, which differs by one component: a new host for a given VM or another temperature for a data center. Regarding TS_1, the new solution is obtained either by moving VM v from server s_1 to server s_2, $M_1(v, s_1, s_2)$, within the same data center, or by changing the temperature of data center d from t_1 to t_2, $M_2(d, t_1, t_2)$. The move with the lowest cost, based on its tabu status and the aspiration criterion, is chosen. To rapidly evaluate the solution, only the cost difference between the actual solution and the neighbor is computed. The gain associated with $M_1(v, s_1, s_2)$ is obtained as follows:

$$G_1(v, s_1, s_2) = \begin{cases} 0 & \text{if} \quad s_1 = s_2 \\ G_1^C + G_1^O + G_1^T + G_1^D + G_1^I & \text{otherwise} \end{cases} \tag{6.23}$$

where G_1^C is the carbon gain, and G_1^O, G_1^T, G_1^D and G_1^I are respectively the penalty gain due to capacity violations, chassis maximal power consumption, performance degradation and interference. The gain associated with $M_2(d, t_1, t_2)$ is given by:

$$G_2(d, t_1, t_2) = \begin{cases} 0 & \text{if} \quad t_1 = t_2 \\ G_2^C + G_2^T & \text{otherwise} \end{cases} \tag{6.24}$$

with G_2^C being the carbon cost gain, and G_2^T being the penalty cost related to the chassis outlet temperature. This approach considerably decreases the execution time, however, at the expense of a more complex implementation. Fig. 6.2 presents the method to select the best move of type 1. Once the best move of each type has been chosen, the one yielding the weakest gain is selected and applied. To avoid returning to visited solutions, two lists of forbidden (tabu) moves, one for each type of move, are implemented: for example, if move $M_1(v, s_1, s_2)$ has been applied, the pair (v, s_1) is added to TL_1, preventing the heuristic from assigning VM v to server s_1 for T_l iterations. The gain tables along with the Tabu lists are updated once the best move has been applied. The exploration stops after K_m iterations without improving the actual best solution or if there is no available move. Also, during the LS process, the solutions are added in a statistics table, which will be passed along to the Diversification mechanism. The LS based on TS_1 is illustrated in Fig. 6.3.

Regarding TS_2, this process tends to efficiently search within the neighborhood of the best solution, by implementing an LS method similar to the previous one, but explores the search space by either permuting VM v_1 and VM v_2, hosted on two different servers (of the same data center), $M_1(s_1, s_2)$, or by swapping temperatures t_1 and t_2 of two different data centers,

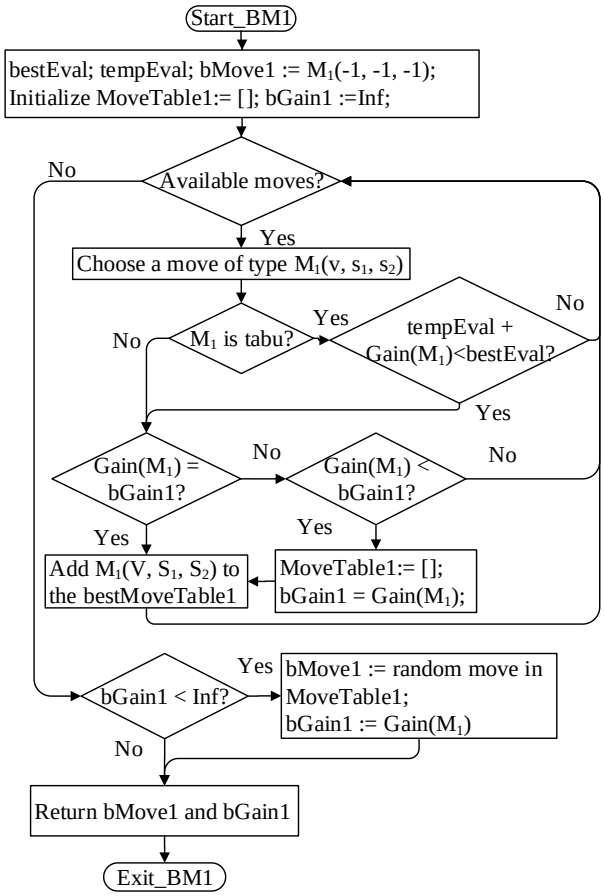


Figure 6.2 Best Move algorithm

$M_2(t_1, t_2)$. For both moves, $M_1(s_1, s_2)$ and $M_2(t_1, t_2)$, two types of gains, $G_1(s_1, s_2)$ and $G_2(t_1, t_2)$ and two tabu lists have also been defined.

6.5.2.3 Perturbation Operator

Although a search history helps avoiding cycles, as the exploration is performed only within a data center, the LS mechanism becomes stagnant after K_m iterations and the perturbation operator is invoked in order to generate a new initial solution. In the perturbation phase, only *pStr%* of the applications and the stand-alone VMs are perturbed, with *pStr* referring to the percentage of a solution components altered by the perturbation. For each chosen VM, it randomly selects a data center, different from the actual host data center, and places the VM on a random server. Regarding an application, a new data center is also chosen, and each application's component (a VM) is randomly assigned to a server located in the selected

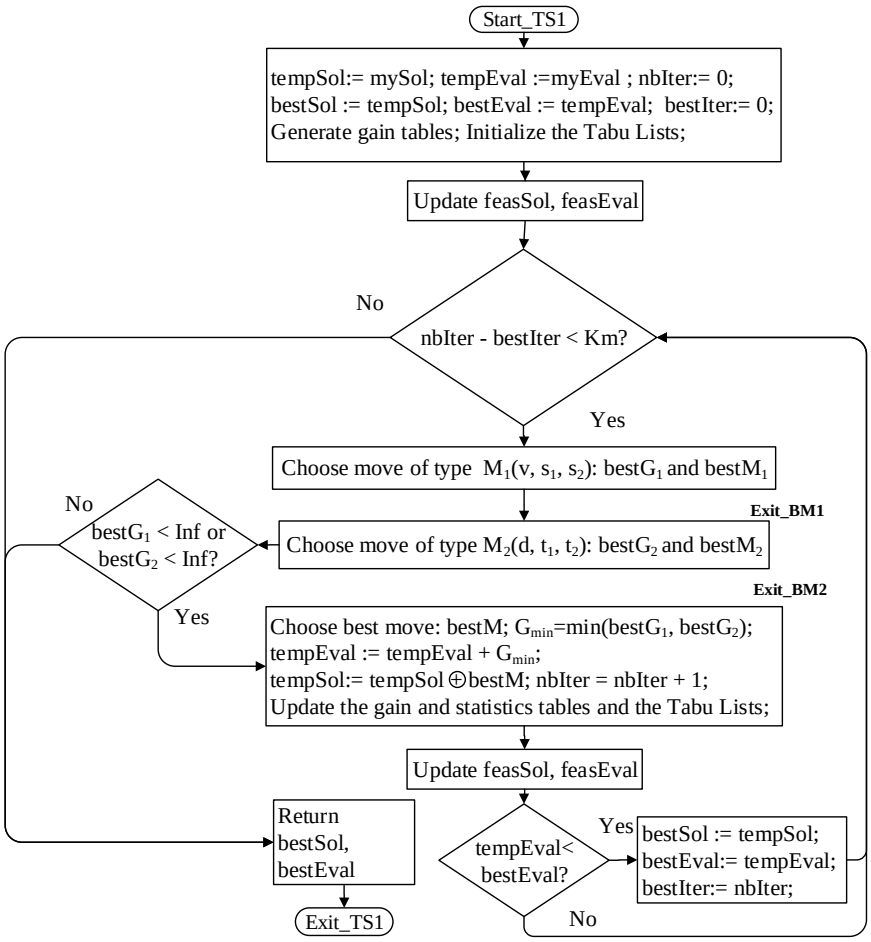


Figure 6.3 TS_1 Mechanism

data center. Once the VMs are placed, the flows are routed.

6.5.2.4 Long-term Memory Mechanism

The Diversification guides the search in so far unexplored regions. Based on the statistics produced from the LS structures, and which refer, for example, to the number of times a VM has been placed on a server, the Diversification process considers the least explored placement pattern to create a new starting point. The same reasoning applies to the applications' host and the data centers' temperature. From this point, a new exploration restarts with a short run of TS_1 followed by the perturbation phase and the Diversification process stops after N_s iterations. The overall ITS_G_{CBF} algorithm is illustrated in Fig. 6.4.

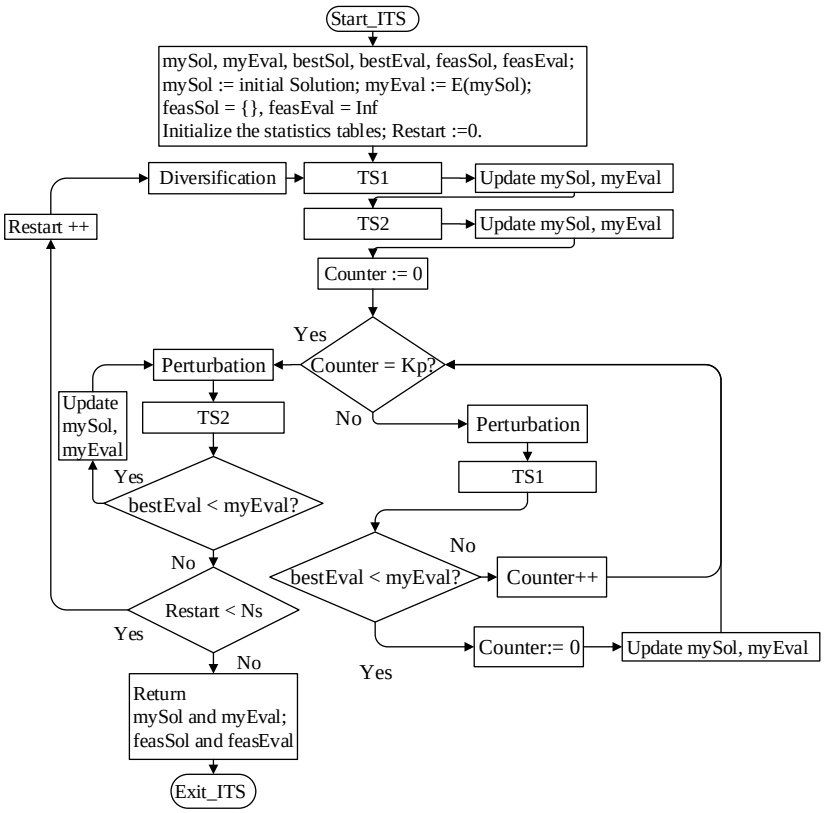


Figure 6.4 Iterated Tabu Search Heuristic

6.6 Computational experiments

In this section, we study the impact of the global $CB_{G}\_OPT\_SLA$ model and the proposed ITS_G_{CBF} algorithm on the deployment of hypothetical workloads in an InterCloud environment.

6.6.1 Experimentation setup

A systematic set of experiments have been conducted considering a physical environment made of up to 10 geographically distributed cloud data centers owned by a unique provider. Each data center contains servers which are organized into a tree architecture, with a single path between each pair of servers. Regarding the workload, we consider a group of up to 5 clients submitting a set of VMs, the associated traffic demands, along with the SLA requirements, to be hosted in the InterCloud environment. For sake of simplicity and without loss of generality, only the constraints related to capacity violation,VMs-VMs interference and

application locations are considered. Regarding VMs compatibility, co-location constraints are neglected in *Scenario 1*, whereas the probability of interference between two VMs is set to 50% in *Scenario 2*. Also, as at any time, only 30% of the VMs are sending and receiving packets (Fang *et al.*, 2013b), we consider the probability of two VMs communicating being set to 50% (partial mesh). Moreover, as for performance purposes, no traffic will be routed outside the data centers, the nodes and links of the backbone network will be ignored. In each experiment, the provider need to decide which data centers should be used, and at which optimal temperature, where the VMs should be placed and how to efficiently route the traffic demands without violating any type of constraints, in order to minimize the total carbon footprint of the InterCloud environment.

In our experiments, the instances are defined by their size, namely the number of data centers D, switches T, chassis X, servers S, VMs V and flows F. The features of the physical infrastructure and the workload are presented in Table 6.1, where the capacity and power consumption profile of each physical node and the resource requirements of each VM and flow can be randomly computed using the minimal and maximal values of each parameter. All the scenarios are run on a Core i5, 3.3 GHz CPU Machine with 8 GB of RAM and running Windows 8. Due to space limitation, results for *Scenario 2* are not presented in this paper.

6.6.2 Comparison with other optimization models

The proposed model is implemented in AMPL/CPLEX, and tested on 9 different cases. In order to demonstrate the contribution of the proposed model, $CB_{G\_}OPT\_SLA$ (A model), the same instance are also tested under three reference models: VM_{OPT} (B model), NET_{OPT} (C model) and EFF_{OPT} (D model). VM_{OPT} optimizes the VM placement in order to reduce their carbon footprint; NET_{OPT} minimizes the network consumption, while EFF_{OPT} optimizes the data centers' power usage effectiveness. Our purpose is to demonstrate the advantages of jointly optimizing VM placement and traffic routing, in the view of minimizing data centers' power consumption and equivalent carbon footprint. The total (TO), the chassis and servers (VM) and the network (NE) carbon footprint costs (in tons), for each model, are provided in Table 6.2. The " * " sign indicates that the instance cannot be solve to optimality, due to resource limitation. Also, the carbon cost gaps of the first eight (8) instances and the total CPU time for each model are illustrated in Fig. 6.5 and Fig. 6.6.

From Table 6.2, it can be seen that none of the comparative models can effectively achieve carbon footprint minimization. With VM_{OPT}, the traffic distribution among the VMs is ignored, which may result in configurations where the network resource usage is poorly optimized. With NET_{OPT}, the VM placement is better aligned with the traffic distribution,

Table 6.1 Parameters

Data centers features	MIN	MAX
Greenness factor ($KgCO_2/1000W$)	100	900
Chassis features		
Number of fans	6	10
Idle power (W)	500	1000
Fan nominal power (W)	5	12
Maximum outlet temperature ($°C$)	93	100
Thermal resistance ($°C/W$)	0.03	1
Switches features		
Idle power (W)	150	170
Traffic power (W)	180	200
Bandwidth capacity (GBs)	10^7	5x10^7
Links features		
Bandwidth capacity (GBs)	10^6	5x10^6
Servers features		
Processor capacity ($CPU - h$)	500	1000
Memory size (GBs)	500	1000
Disk capacity (GBs)	500	1000
Idle power (W)	60	100
Processor power (W)	70	240
Disk power (W)	50	280
VMs features		
Processor requirements ($CPU - h$)	25	75
Memory size (GBs)	25	75
Disk capacity (GBs)	25	75
Flows features		
Bandwidth requirements (GBs)	5000	10000

Table 6.2 Carbon footprint cost comparison with baseline models

Inst. set	Problem size	$CB_{G}\_OPT\_SLA$			VM_{OPT}			NET_{OPT}			EFF_{OPT}		
	(D,T,X,S,V,F)	TO	VM	NE	TO	VM	NE	TO	VM	NE	TO	VM	NE
		(t)	(t)	(t)	(t)	(t)	(t)	(t)	(t)	(t)	(t)	(t)	(t)
1	1, 9, 4, 8, 20, 46	2.61	2.44	0.17	4.02	2.44	1.58	7.23	7.06	0.17	8.62	7.06	1.56
2	1, 9, 4, 8, 30, 106	4.51	4.16	0.35	5.66	4.06	1.60	7.61	7.27	0.34	8.82	7.27	1.56
3	1, 9, 4, 8, 40, 190	5.04	4.70	0.34	6.30	4.70	1.60	8.13	7.79	0.34	9.35	7.79	1.56
4	2, 18, 8, 16, 20, 46	1.56	1.47	0.09	4.02	1.47	2.56	10.01	9.92	0.09	6.34	3.79	2.55
5	2, 18, 8, 16, 30, 106	2.30	2.12	0.19	4.47	2.02	2.45	10.08	9.90	0.18	6.45	3.90	2.55
6	2, 18, 8, 16, 40, 190	2.70	2.51	0.19	*	*	*	10.38	10.19	0.18	6.84	4.30	2.55
7	3, 27, 12, 24, 20, 46	1.15	1.03	0.12	4.18	1.01	3.17	11.18	11.13	0.05	5.10	1.97	3.13
8	3, 27, 12, 24, 30, 106	1.29	1.11	0.18	4.11	1.11	3.00	11.43	11.28	0.15	5.22	2.09	3.13
9	3, 27, 12, 24, 40, 190	*	*	*	4.67	1.60	3.06	11.86	11.67	0.18	5.45	2.32	3.13

however, not only the chassis and servers footprint is not considered, but the complex interactions between the cooling and the dynamic behavior of the fans is not captured by the model. Regarding EFF_{OPT}, it minimizes the cooling overhead with respect to the equipment power consumption, without necessarily optimizes the latter. Regarding the cost gaps, although EFF_{OPT} simultaneously considers VM placement and traffic routing, as $CB_{G}\_OPT\_SLA$,

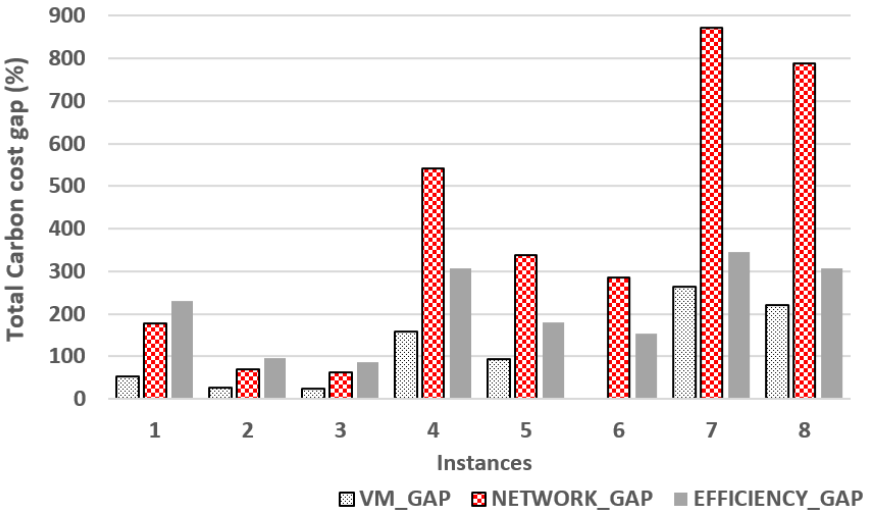


Figure 6.5 Carbon footprint cost gap

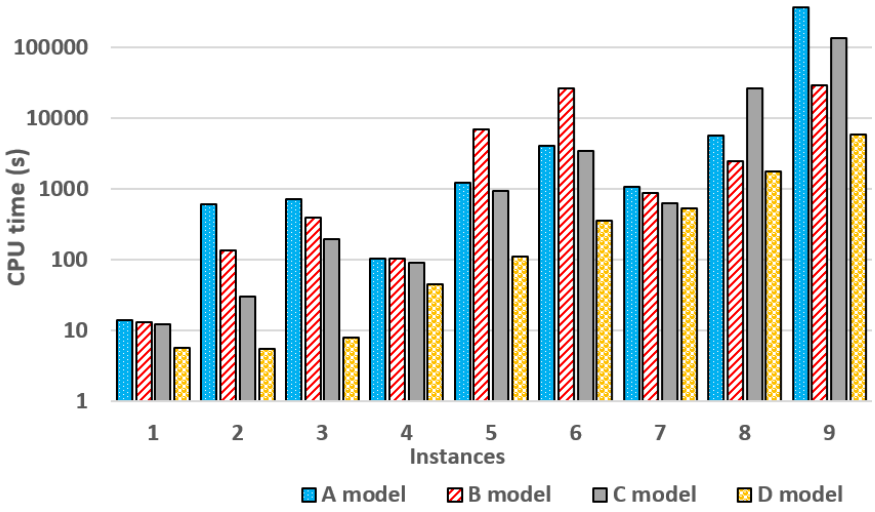


Figure 6.6 Time comparison

VM_{OPT} consistently performs better, as depicted in Fig. 6.5, mainly because it takes into account the dynamic behavior of the server fans against the gain of the CRAC at high temperatures. Also, as the reference models might be conflicting goals, $CB_G\_OPT\_SLA$ cannot always optimize the three objectives separately, but can jointly considers them in order to achieve an overall carbon footprint optimization. These results help assessing the efficiency of the proposed model, as it considers a joint optimization of the computing nodes power consumption, including their dynamic behavior, the network resources dissipation and the cooling efficiency. This model provides a more accurate evaluation of the carbon footprint

cost, allowing one to perform an efficient VM placement, however at the expense of a longer average execution time, as illustrated in Fig.6.6, due to the high complexity of the problem.

6.6.3 ITS_G_{CBF} Performance Evaluation

In this section, the efficiency of the proposed algorithm, in terms of carbon cost and computing time, is evaluated through extensive simulations: ITS_G_{CBF} is first compared to a lower bound value, for small instances, then to other baseline algorithms for bigger problem sizes. Preliminary testing helped identify the optimal parameter for each mechanism, where most of them depend on the problem size. The parameters used in ITS_G_{CBF} are as follows: $K_m = 5 * \sqrt{V + S + D + \Lambda}$, where Λ is the number of temperature values, $K_p = 2.5 * K_m$, $pStr = 25\%$ and $N_s = \sqrt{S + \Lambda}/8$, for small instances, and $K_m = \sqrt{V + S + D + \Lambda}$ and $K_p = K_m/8$ for bigger problem sizes. The number of iterations, K_m and K_p are reduced for large problems due to the fact that the cost improvement, in terms of carbon footprint, is significantly low compared to the increase in the CPU time associated with high number of iterations. The number of iterations to keep a move in a Tabu list is randomly determined, at each iteration of the LS process, using these intervals: $\sqrt{V * S/D} * [0.75 - 1]$ for the first move type, and $\sqrt{D * \Lambda} * [0.75 - 1]$ for the move of type 2. ITS_G_{CBF} and the reference resolution methods are implemented in C++ and each case are executed 10 times.

6.6.3.1 ITS_G_{CBF} and Lower Bound

The proposed ITS_G_{CBF} heuristic is executed on 20 small instances and the results are compared with the optimal solution, OPT_G_{CBF}, computed using a linear program implemented in AMPL/CPLEX. The resulted lower bound enables to assess the performance of the proposed resolution method. The results are presented in Table 6.3, where column 2 and 3 give respectively the problem sizes and the carbon footprint cost calculated with the exact method. Regarding ITS_G_{CBF}, the average carbon footprint and the cost gaps (minimal, mean and maximal) with respect to the lower bound values are shown in columns 4 to 7. The total number of iterations as well as the iteration at which the near-optimal solution is found are also presented in the last two columns.

Table 6.3 demonstrates that, in general, the carbon footprint costs obtained from the implementation of ITS_G_{CBF} are very close to the lower bound values. Also, as illustrated by the results, the proposed method is always able to find the optimal solution in instances 1, 2, 3, 14, 15, 16 and 18, as a minimal, average and maximal distance to the lower bound of 0% is observed. Although most of the gaps are below 3%, which helps in showing that ITS_G_{CBF} is very effective in determining near-optimal solutions, it can be observed that

Table 6.3 Comparison between OPT_G_{CBF} and ITS_G_{CBF}

Inst. sets	Problem size (D, T, X, S, V, F)	OPT_G_{CBF}	ITS_G_{CBF}					
		Carb. Foot. ($KgCO_2$)	Carb. Foot. ($KgCO_2$)	Cost gaps Min (%)	Mean (%)	Max (%)	Total Iter. #	Best Iter. #
1	1, 11, 6, 18, 60, 34	4270.75	4270.75	0.00	0.00	0.00	18204	1136
2	1, 11, 6, 30, 60, 34	3227.42	3227.42	0.00	0.00	0.00	21657	3084
3	1, 11, 6, 30, 76, 31	5043.47	5043.47	0.00	0.00	0.00	23601	803
4	1, 15, 10, 30, 60, 34	5050.87	5119.50	0.34	1.36	2.99	22792	3506
5	1, 15, 10, 50, 60, 34	3217.98	3297.38	2.47	2.47	2.47	39738	483
6	1, 9, 4, 12, 56, 24	11131.66	11172.19	0.34	0.36	0.38	41975	25226
7	1, 9, 4, 16, 56, 24	9377.61	10384.77	10.35	10.74	11.67	42350	28351
8	2, 22, 12, 60, 60, 34	2972.66	3009.69	0.04	1.25	3.06	38270	71
9	2, 30, 20, 100, 60, 34	2740.07	2761.56	0.78	0.78	0.78	54550	59
10	2, 18, 8, 24, 60, 30	12328.54	12387.59	0.25	0.48	0.58	27782	10092
11	2, 22, 12, 36, 46, 32	2937.75	3478.47	18.31	18.41	18.45	18046	967
12	2, 22, 12, 36, 52, 32	3050.41	3072.01	0.38	0.71	1.63	23834	9456
13	2, 22, 12, 60, 46, 32	2742.17	2785.22	0.67	1.57	2.12	28018	224
14	2, 22, 12, 60, 52, 32	2917.44	2917.43	0.00	0.00	0.00	32229	21
15	3, 33, 18, 54, 60, 34	2566.64	2566.64	0.00	0.00	0.00	35389	6
16	3, 33, 18, 54, 46, 32	2437.97	2437.97	0.00	0.00	0.00	27343	12
17	3, 33, 18, 54, 52, 32	2641.56	2644.30	0.10	0.10	0.10	33219	32
18	3, 33, 18, 90, 46, 32	2437.97	2437.97	0.00	0.00	0.00	35251	12
19	3, 33, 18, 90, 52, 32	2641.56	2644.30	0.10	0.10	0.10	41314	33
20	4, 44, 24, 72, 51, 45	1953.49	2016.75	1.08	3.24	5.94	50481	23521

the cost obtained in instance 11 may reach a maximal distance of about 18.45% with respect to the optimal value. This is due to the fact that this search space is more difficult to explore with the defined parameters, as the heuristic is trapped in a local minimum and is not able to improve the solution, during the following iterations. For this particular test instance, the simulations are extended by considering a perturbation strength of 50% while keeping the other parameters unchanged. The obtained results show that the resolution method was able to reduce the carbon footprint to an average cost of 2999.73 $KgCO_2$, compared to 3478.47 $KgCO_2$, and a maximal gap of about 3.6%. This helps in demonstrating that for some cases such as instance 11, when the jump is too small, new regions of the search space cannot be reached and the impact of the perturbation is rapidly negate by the LS process.

As the proposed resolution method is implemented in order to allow a good tradeoff between the CPU time and the quality of the solutions, the execution time of ITS_G_{CBF} under the tested instances of Table 6.3 is also analyzed and compared with the CPU time obtained with the exact method. More specifically, Fig. 6.7 depicts the minimal, average and maximal CPU time resulted from the implementation of the proposed resolution method, as well as the

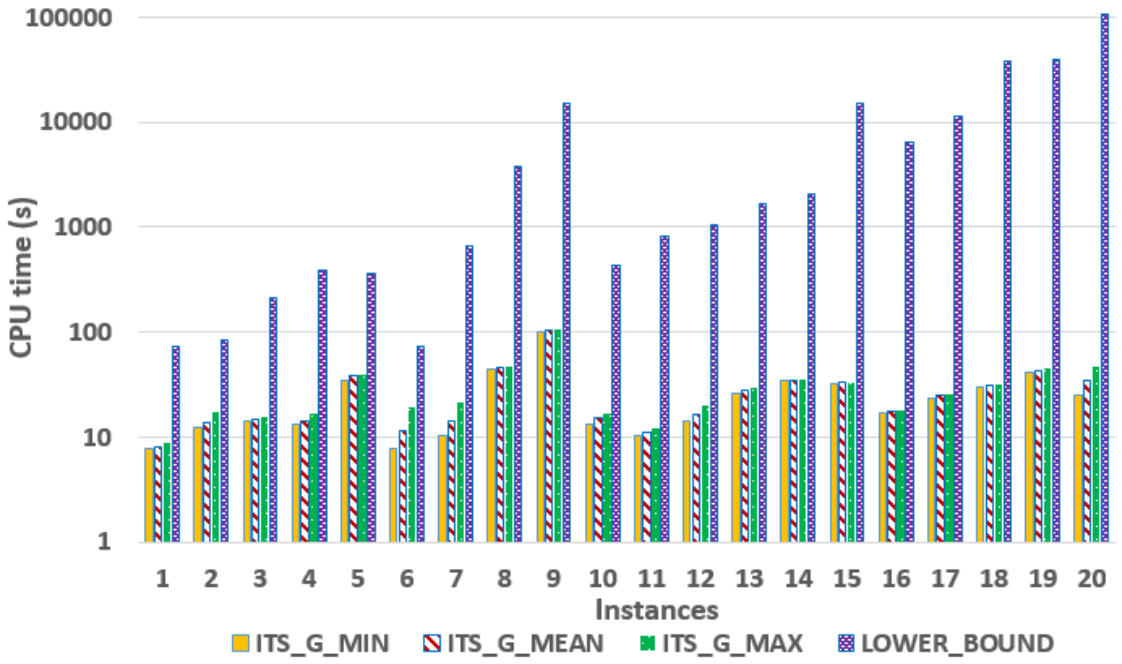


Figure 6.7 Time comparison between ITS_G_{CBF} and OPT_G_{CBF}

execution time of CPLEX. It can be seen that the execution time for ITS_G_{CBF} ranges from about 7.69 to 107.45 seconds, whereas OPT_G_{CBF} takes close to 30 hours (107804 seconds) to solve the hardest problem (instance 20). In general, computational results demonstrate that the running time for ITS_G_{CBF} increases linearly, with respect to the problem size, while the CPU time for the exact method OPT_G_{CBF} follows an exponential growth based on the length of the input.

Results from Table 6.3 and Fig. 6.7 clearly exhibit the efficiency of the proposed method. The chosen parameters enable OPT_G_{CBF} to suit multiple problem sizes, by finding the optimal solution at the early stage of the exploration, as in instances 14, 15, 16 and 18, or by efficiently exploring more difficult search spaces in order to determine the near-optimal solution, as in instances 4, 6, 10, 12 and 20. More specifically, with an average maximal carbon footprint gap, among all the tested instances, of about 2.51%, and an average computing time less than 105 seconds, the results demonstrate that the proposed approach outperforms OPT_G_{CBF}, by allowing a good tradeoff between the quality of the solution and the computational time.

6.6.3.2 ITS_G_{CBF} and large cases

In order to gauge the performance of the proposed resolution method, ITS_G_{CBF} is executed on problems that could not be solved to optimality using the AMPL/CPLEX program

and the results are compared with those obtained with several baseline approaches: a random assignment (*RAND*) and three variants of the proposed heuristic, namely *ILS*, *TS* and *ITS_RAND*. In *RAND*, VMs and applications are randomly placed, and the data centers' temperature are randomly determined. The first variant, *ILS*, is ITS_G_{CBF} with a descent operator as the LS process and without the Diversification mechanism, *TS* implements the first LS process TS_1 and in *ITS_RAND*, the initial configuration is generated using a random placement.

The proposed ITS_G_{CBF} heuristic as well as the reference resolution methods are run on twelve (12) different instances and the results for the ITS_G_{CBF} algorithm, regarding average carbon footprint cost, total CPU time and solution-time (time at which the best solution is found) are provided in Table 6.4. The last four (4) columns exhibit the average carbon footprint cost resulted from the implementation of the comparative methods. From the results in Table 6.4, it can be observed that, in terms of carbon footprint cost, ITS_G_{CBF} always outperforms the reference algorithms. The carbon footprint difference is more noticeable with the *RAND* approach where the placement is randomly performed with no further exploration of the search space in order to improve the carbon cost. In this context, this approach may end up in poor configurations, with an extremely high carbon footprint cost, or may even result in non-feasible configurations, as illustrated in instances 1 to 4, where from 10 to 100% of the obtained solutions where unfeasible. Also, even though the carbon gaps obtained with the first two (2) variants of ITS_G_{CBF}, namely *ILS* and *TS* are small, the proposed approach always performs better when the perturbation phase of the *ILS* process is combined with the memory mechanism of the LS approach. Also, it can be observed that *TS* is the best comparative algorithm, mainly because it implements a Tabu list that avoids cycles and local optima. Regarding the last reference method, *ITS_RAND*, although it explores the search space with a perturbation and an LS operator such as in ITS_G_{CBF}, but from an initial random placement, it always results in a bad configuration with a high carbon cost. This is due to the fact that, although enhanced methods, such as a perturbation phase or memory mechanisms are implemented, as the search starts from a random point, where no guarantee can be made regarding the quality and the feasibility of the initial solution, it becomes harder to efficiently explore the neighborhood and obtain good solutions, as the algorithm may be trapped in bad local optima. The cost difference can also be seen in Fig. 6.8, which presents the best solution found by this latter reference approach, compared to the proposed method. From this figure, we observe that the cost gap increases, in general, with respect to the problem size. Therefore, not only ITS_G_{CBF} provides the best solution, but it is also more scalable than the *ITS_RAND* approach.

As the first three (3) comparative algorithms are fast heuristics, mainly because they im-

Table 6.4 Carbon footprint and CPU time comparison between ITS_G_{CBF} and other reference methods

Inst. set	Problem size (D, T, X, S, V, F)	ITS_G_{CBF}			*RAND*	*ILS*	*TS*	*ITS_RAND*
		Carb. Foot. (t)	CPU (s)	CPU-sol (s)	Carb. Foot. (t)	Carb. Foot. (t)	Carb. Foot. (t)	Carb. Foot. (t)
1	1, 15, 10, 50, 430, 931	25.31	204.82	44.76	997090.90	25.53	25.49	28.32
2	1, 15, 10, 50, 550, 1151	31.34	430.94	101.75	3113773.00	31.41	31.40	34.30
3	2, 30, 20, 100, 500, 929	16.91	99.06	56.42	68.90	17.03	17.03	22.21
4	2, 30, 20, 100, 500, 1021	15.37	114.38	37.79	37.30	15.41	15.42	21.91
5	4, 60, 40, 200, 580, 805	11.33	92.65	34.86	44.82	11.34	11.34	21.11
6	4, 60, 40, 200, 580, 1057	11.84	141.47	11.63	45.61	11.85	11.85	22.06
7	6, 90, 60, 300, 560, 817	9.29	143.16	24.27	69.04	9.30	9.30	30.36
8	6, 90, 60, 300, 770, 1189	15.30	367.85	123.25	73.82	15.35	15.35	37.67
9	8, 120, 80, 400, 660, 805	12.90	207.21	63.28	103.33	12.91	12.91	41.85
10	8, 120, 80, 400, 770, 931	13.70	339.33	102.77	105.43	13.74	13.74	44.66
11	10, 150, 100, 500, 860, 1026	16.17	432.68	48.63	124.90	16.29	16.29	50.93
12	10, 150, 100, 500, 890, 1007	14.63	329.65	16.41	124.42	14.63	14.63	51.53

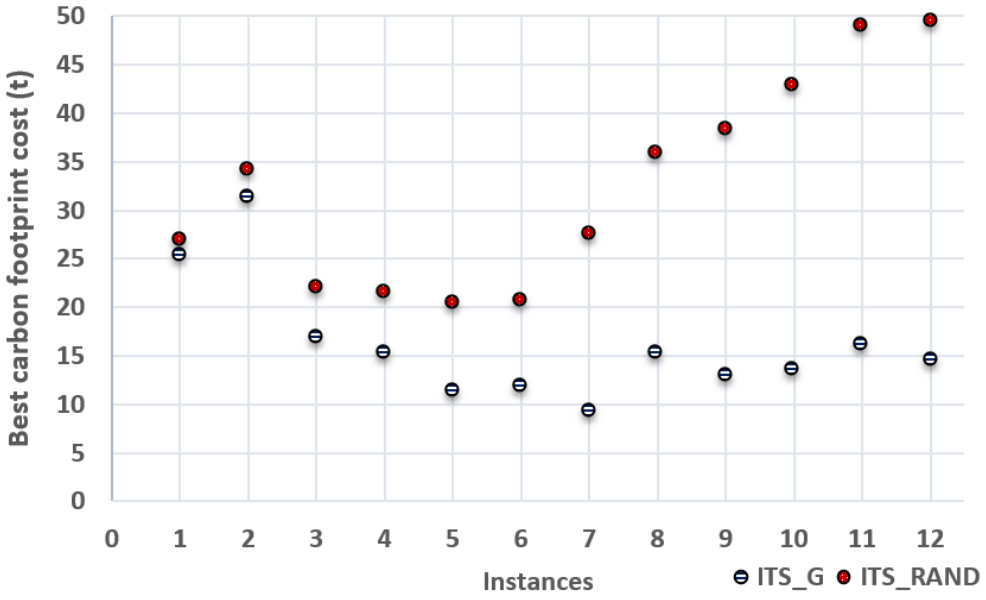


Figure 6.8 Best cost comparison between ITS_G_{CBF} and ITS_RAND

plement a single mechanism, the performance of the proposed approach, in terms of computational time, is only compared to the execution time of the *ITS_RAND* method. In that context, the total execution time (minimal, average and maximal) of each algorithm is illustrated in Fig. 6.9, whereas their best solution-time is depicted in Fig. 6.10. From these figures, it can be seen that, compared to the *ITS_RAND* method, the proposed approach

always yields the best performance, not only in terms of carbon footprint cost, as illustrated in Fig. 6.8, but also in terms of computational time. Intuitively, a high carbon footprint cost would suggest a short exploration time, and a long execution time would normally lead to low-cost configurations. However, as illustrated in Fig. 6.8 and Fig. 6.10, not only the best solution found by *ITS_RAND* is of bad quality, but the CPU time needed to obtained that solution is extremely high, for most of the tested instances. This has an impact on the total execution time of the comparative algorithm, as presented in Fig. 6.9, where the CPU time can be as close to 600 seconds, whereas the maximum computational time observed for the proposed algorithm is around 470 seconds. These results exhibit the poor behavior of the *ITS_RAND* compared to the proposed approach, as the latter is able to provide low-cost solutions in a shorter period of time.

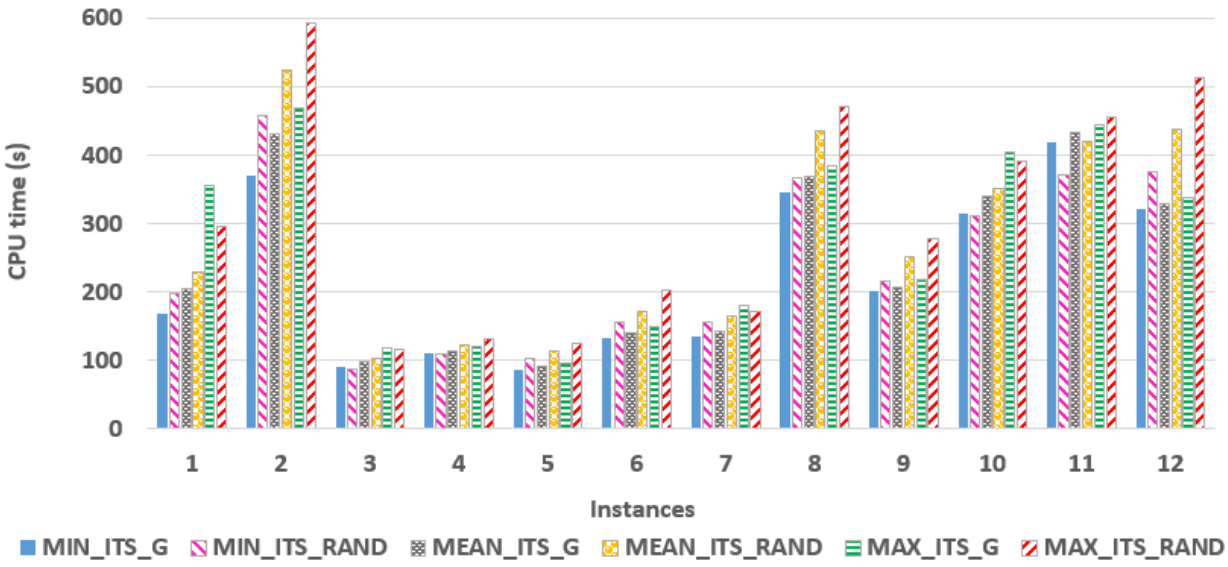


Figure 6.9 Time comparison between ITS_G_{CBF} and *ITS_RAND*

The obtained results help assessing the efficiency of the proposed ITS_G_{CBF} algorithms, both in terms of carbon footprint cost and CPU time. As an NP-hard problem, the workload placement problem needs to be handled with attention, especially when joint optimization is performed. In that sense, it has been proven that a random placement may result in poor configurations where the percentage of feasible solutions is low. Moreover, although the gaps observed between ITS_G_{CBF} and its variants *ILS* and *TS* are small for large-sized problems, we observe that the use of both operators allow ITS_G_{CBF} to reach near-optimal solution for small instances, within a reasonable amount of time. Furthermore, besides the perturbation and the LS operators, the generation of the initial configuration needs to be well designed,

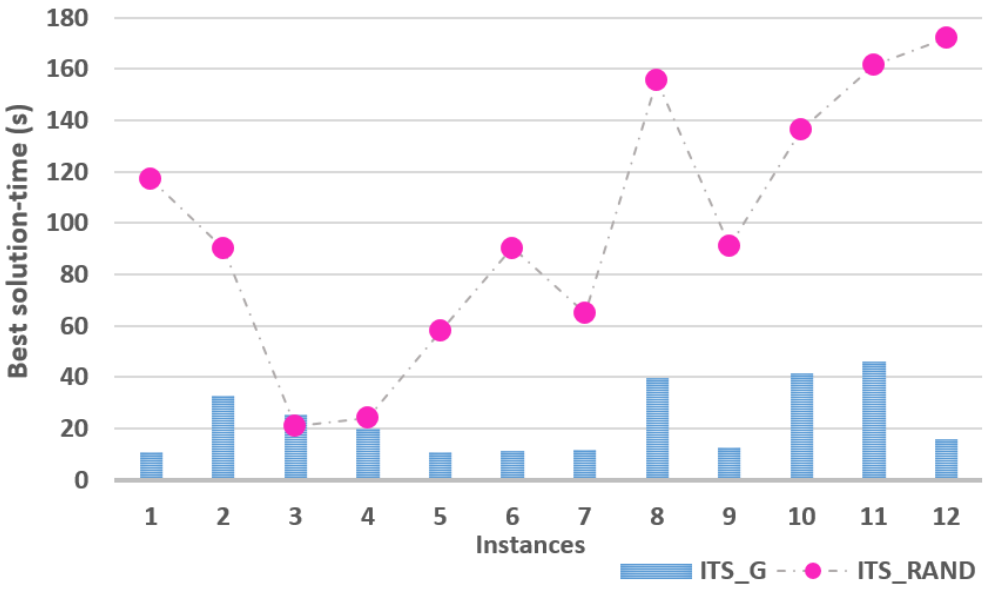


Figure 6.10 Best solution-time comparison between ITS_G_{CBF} and ITS_RAND

as a bad starting point may hinder the search process, even with enhanced methods such as a perturbation phase or memory mechanisms, and leads the exploration to cycle around bad local optima. Based on this analysis, these results helped in proving that the use of the proposed ITS_G_{CBF} is what effectively solves the workload placement problem, as the algorithm has a higher flexibility to explore the search space and provides a good tradeoff between the quality of the solution and the computing time.

6.7 Conclusion

In this paper, we proposed a new mathematical model, that jointly performs VM placement and traffic routing, with the view of minimizing the carbon footprint of an InterCloud environment. As the workload placement is classified as an NP-hard problem, we also proposed an implementation of an hybrid metaheuristic, ITS_G_{CBF}, that combines the perturbation operator of the ILS heuristic to the short-term memory mechanism of the TS algorithm. This heuristic has been further improved by using a second LS process, also based on the TS heuristic, as well as a Diversification process that enables to explore unvisited regions.

Preliminary experiments help determining the optimal parameters for each phase of the ITS_G_{CBF}, which have been later used to evaluate the performance of the proposed approach. Comparisons, for small instances, with the exact method demonstrate that ITS_G_{CBF} was able to reach optimal or near-optimal solutions with an overall maximal cost gap of about

2.51%, in less than 110 seconds, compared to CPLEX, where the computing time may be as close as 30 hours. Results issued from the comparison with other reference methods, exhibit the efficiency of the proposed method in solving large-sized problems as it is able to determine good configuration, in less than 500 seconds, which configuration may lead to carbon footprint savings up to 240%, providing therefore an excellent tradeoff between the quality of the solutions and the computational time.

We end this paper by motivating the use of efficient techniques, such as heat redistribution or the free cooling mechanisms, in order to improve the CRAC efficiency and lower the total carbon footprint cost. The heat recirculation can be also integrated into an extended version of the model, as physical separators are not always available inside a data center. Furthermore, as the workload evolves over the time, migration techniques could also be considered in order to reflect the workload placement in real cloud scenario. Regarding the resolution method, the combination of different perturbation mechanisms as well as multiple LS operators can be of great interest in order to improve the efficiency of the hybrid algorithm.

CHAPITRE 7 RÉSULTATS COMPLÉMENTAIRES

Dans cette section, nous présentons quelques résultats qui n'ont pas pu être illustrés dans les trois travaux précédents.

7.1 Résultats complémentaires du volet 1

Dans le premier volet de notre travail, nous avons présenté un modèle de placement des applications à VM unique, visant à optimiser l'empreinte carbone d'un environnement InterCloud, sous la contrainte de différents requis de performances. Toutefois, seules les contraintes de co-localisation ont été prises en compte dans les résultats présentés au chapitre 4. Aussi, cette section a pour but d'illustrer les performances du modèle proposé, *CB_OPT_SLA*, par rapport à un modèle d'empreinte carbone, *CB_OPT*, ne respectant que les contraintes de capacité physique des équipements.

7.1.1 Contraintes de température de sortie des châssis

Afin d'illustrer le comportement du modèle proposé face aux contraintes de température de sortie des châssis, nous avons considéré les mêmes tailles d'instances que celles du Tableau 4.7, mais avons modifié les caractéristiques des équipements afin de pouvoir illustrer la violation des contraintes liées à la température de sortie des châssis. Aussi, les tailles de jeux de données du Tableau 4.7 ne pouvant être résolues optimalement avec CPLEX, n'ont pas été considérées.

Dans un premier temps, les performances du modèle sont évaluées en termes de coût d'empreinte carbone. Aussi, pour illustrer les mérites du modèle sans contraintes, à réaliser une consolidation aveugle, nous avons également considéré le coefficient CF qui se définit comme le rapport entre le taux de violation de contraintes, sur le pourcentage de gain d'empreinte carbone. Pour ce qui est des contraintes liées à la température, le taux de violation s'évalue par le rapport du nombre de châssis dont la température de sortie dépasse la limite permise, sur le nombre de châssis actifs. Les résultats issus de cette comparaison sont présentés dans le Tableau 7.1, où l'empreinte carbone résultant de l'optimisation des 2 modèles est illustrée dans les colonnes 3 et 4, et les valeurs de coefficient liées aux contraintes de co-localisation et de température, dans les deux colonnes suivantes.

Comme nous nous y attendions, l'empreinte carbone découlant du modèle sans contrainte est toujours inférieure à celle obtenue par le modèle proposé, car, en dehors des limites physiques

Tableau 7.1 Empreinte carbone et valeurs de coefficients de fitness

Instances	InterCloud (dimension)	*CB_OPT_SLA*	*CB_OPT*		
	(D,X,S,V)	Emp. Carb. ($KgCO_2$)	Emp. Carb. ($KgCO_2$)	CF_Co	CF_Temp
1	1, 4, 20, 20	1495.48	1389.76	14.15	0
2	1, 4, 20, 40	1580.93	1496.15	18.65	0
3	1, 4, 40, 40	1580.93	1496.15	18.65	0
4	1, 4, 40, 80	1818.55	1818.55	Inf	***
5	1, 4, 60, 60	1707.41	1707.41	Inf	***
6	1, 4, 60, 120	2131.16	2131.16	Inf	***
7	1, 8, 40, 40	1580.93	1496.15	18.65	0
8	1, 8, 40, 80	1818.55	1818.55	Inf	***
9	1, 8, 80, 80	1818.55	1818.55	Inf	***
10	1, 12, 60, 60	1639.05	1605.85	29.21	49.38
11	1,12, 60, 120	2394.89	1982.17	1.97	5.80
12	2, 8, 40, 20	1090.23	1031.93	18.70	0
13	2, 8, 40, 40	1140.01	1081.71	19.55	0
14	2, 8, 80, 40	1140.01	1081.71	19.55	0
15	2, 8, 80, 80	1295.86	1295.86	Inf	***
16	2, 8, 120, 60	1225.59	1225.59	Inf	***
17	2, 8, 120, 120	1504.33	1504.33	Inf	***
18	2, 16, 80, 40	1140.01	1081.71	19.55	0
19	2, 16, 80, 80	1295.86	1295.86	Inf	0
20	2, 16, 160, 80	1295.86	1295.86	Inf	***
21	2, 24, 120, 60	191.98	191.98	Inf	***
22	2, 24, 120, 120	238.14	238.14	Inf	***
23	3, 12, 60, 20	1090.23	855.93	4.65	4.65
24	3, 12, 60, 40	1140.01	971.02	3.17	6.75
25	3, 12, 120, 40	1140.01	971.02	3.21	6.75

des équipements, le modèle *CB_OPT* ne considère aucune autre contrainte limitant la consolidation des VMs, ce qui résulte en un nombre moins élevé d'équipements actifs que dans le cas du modèle proposé. Toutefois, ces résultats deviennent moins intéressants lorsqu'on analyse les valeurs de CF. En effet, pour les jeux de données considérés, pour ce qui est du CF associé aux contraintes de co-localisation, CF_Co, on observe qu'à l'instar des résultats présentés au chapitre 4, les valeurs sont en général excessivement élevées, témoignant d'une consolidation inefficace. Aussi, le modèle proposé est d'autant plus efficace, car, dans certains cas, il est en mesure de déterminer un schéma de configuration résultant en un coût d'empreinte carbone identique à celui obtenu avec le modèle *CB_OPT*, mais avec en plus le mérite de respecter toutes les contraintes du problème. En ce qui a trait au CF associé aux limites de température, non seulement les violations sont beaucoup moins fréquentes, mais on observe que pour certaines instances, les deux modèles obtiennent les mêmes résultats, en termes

de coût d'empreinte carbone et de performance. Pour ces résultats, le CF_Temp associé est remplacé par le symbole "***". Toutefois, il existe certains jeux de données où ce coefficient est excessivement élevé, comme en témoigne l'instance 10 où la valeur de CF_Temp s'évalue à environ 49.38. Aussi, comme il n'existe aucune garantie en ce qui a trait à la performance du modèle sans contraintes, il demeure préférable d'utiliser le modèle proposé afin de minimiser l'empreinte carbone sous la contrainte de divers requis de performances.

Les deux modèles ont également été comparés au niveau du temps d'exécution, comme l'illustre la Figure 7.1. En raison de la grande complexité du modèle proposé, le temps de convergence vers la solution optimale est beaucoup plus élevé.

7.1.2 Contraintes de performances

La dernière étape d'évaluation du modèle *CB_OPT_SLA* consiste à intégrer les contraintes relatives à la dégradation de performance des applications. À cette phase, les caractéristiques des demandes ont été modifiées car les premières instances utilisées ne permettaient pas d'illustrer le phénomène de dégradation de performance des applications co-hébergées. Une approche similaire à celle présentée à la section précédente a été considérée. L'empreinte carbone résultant des 2 modèles est présentée et, pour éviter une certaine redondance, seules les valeurs de CF découlant des violations de contraintes de performances sont illustrées dans le Tableau 7.2. Plus précisément, CF_DISK évalue le taux de dégradation liée aux

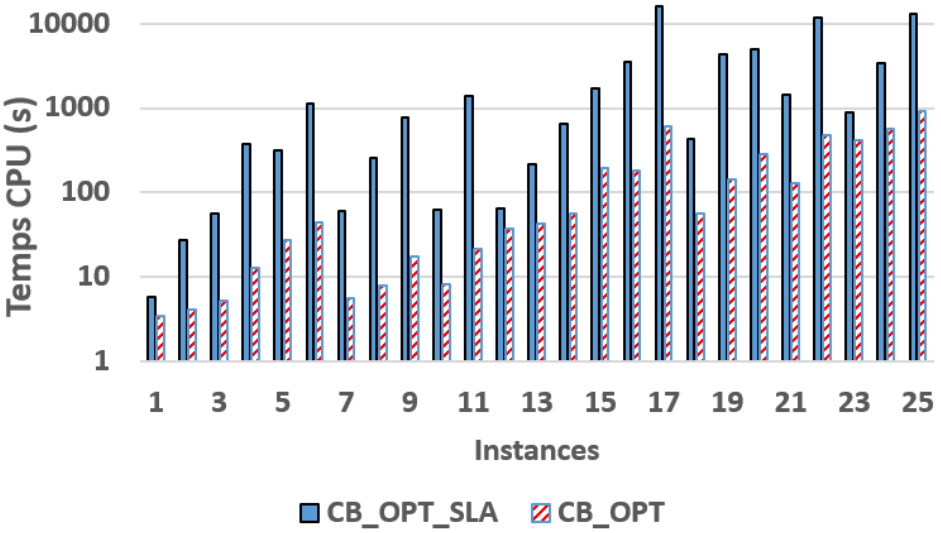


Figure 7.1 Temps CPU pour les modèles avec et sans contraintes (co-localisation et température de sortie)

Tableau 7.2 Empreinte carbone et performances (toutes les contraintes)

Instances	InterCloud (dimension)	*CB_OPT_SLA*	*CB_OPT*		
	(D, X, S, V)	Emp. Carb. ($KgCO_2$)	Emp. Carb. ($KgCO_2$)	CF_DISK	CF_MIXED
1	1, 4, 20, 20	3383.96	3383.96	***	***
2	1, 4, 20, 40	RL	6677.99	-	-
3	1, 4, 40, 40	RL	5670.38	-	-
4	1, 4, 40, 80	RL	12660.78	-	-
5	1, 4, 60, 60	RL	9227.66	-	-
6	1, 8, 40, 40	RL	6677.99	-	-
7	2, 8, 40, 20	523.59	403.73	0.00	1.75
8	2, 8, 40, 40	794.59	794.59	Inf	Inf
9	2, 8, 80, 40	674.54	674.54	Inf	Inf
10	2, 8, 80, 80	RL	1525.31	-	-
11	2, 16, 80, 40	RL	794.59	-	-
12	2, 16, 80, 80	RL	1772.37	-	-
13	3, 12, 60, 20	RL	403.73	-	-
14	3, 12, 60, 40	794.59	794.59	***	***
15	3, 12, 120, 40	674.54	674.54	***	***

applications à forte consommation de disque, et CF_MIXED représente la violation relative à la consolidation de charge mixte. De plus, les instances qui n'ont pas pu être résolues de manière exacte avec au moins l'une des deux méthodes n'ont pas été considérées.

Comme l'illustre la colonne 3 du Tableau 7.2, en raison de la complexité du modèle d'empreinte carbone intégrant toutes les contraintes du problème, plus de la moitié des instances n'ont pas pu être résolues optimalement avec CPLEX. En ce sens, les coefficients de fitness correspondants n'ont pas pu être déterminés comme l'indique le symbole "-" aux colonnes 5 et 6. Nous pouvons également observer que dans certains cas, comme pour les instances 1, 14 et 15, les deux modèles parviennent à la même configuration, avec un coût d'empreinte carbone minimal et aucune violation de contrainte. Toutefois, dans le cas des instances 7, 8 et 9, le comportement de *CB_OPT* est moins intéressant, car il conduit à des solutions où le coefficient de fitness est supérieur à 1, ou peut même tendre vers l'infini, ce qui témoigne d'une mauvaise assignation de la charge. Aussi, afin de mieux illustrer l'inaptitude du modèle *CB_OPT* à réaliser une consolidation efficace des VMs, particulièrement dans les cas où les coefficients de fitness n'ont pas pu être calculés, nous avons également illustré, à la Figure 7.2, le taux de violation de performances. Ainsi, nous pouvons noter que dans bien des situations, *CB_OPT* parvient à respecter toutes les contraintes de performances des applications hébergées. Toutefois, il n'existe aucune garantie quant à l'efficacité du modèle car, pour environ la moitié des instances considérées, nous observons un taux de violation des contraintes non

négligeable et pouvant s'élever jusqu'à environ 60%.

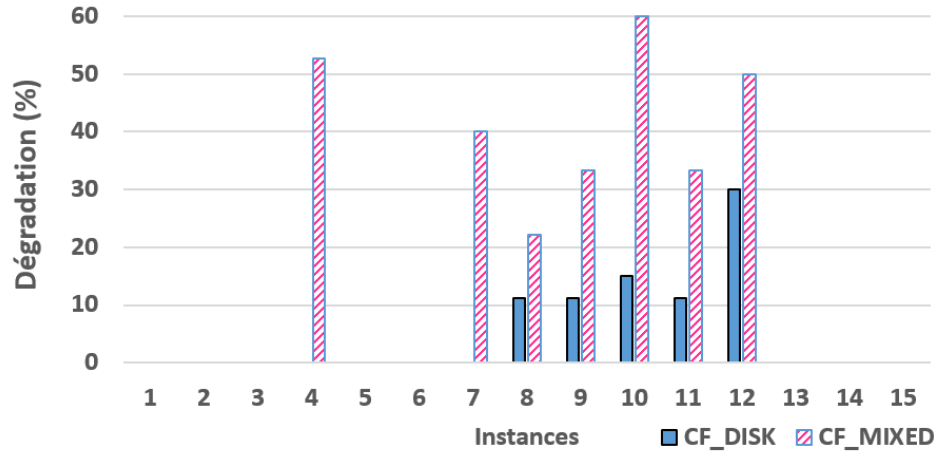


Figure 7.2 Pourcentage de dégradation

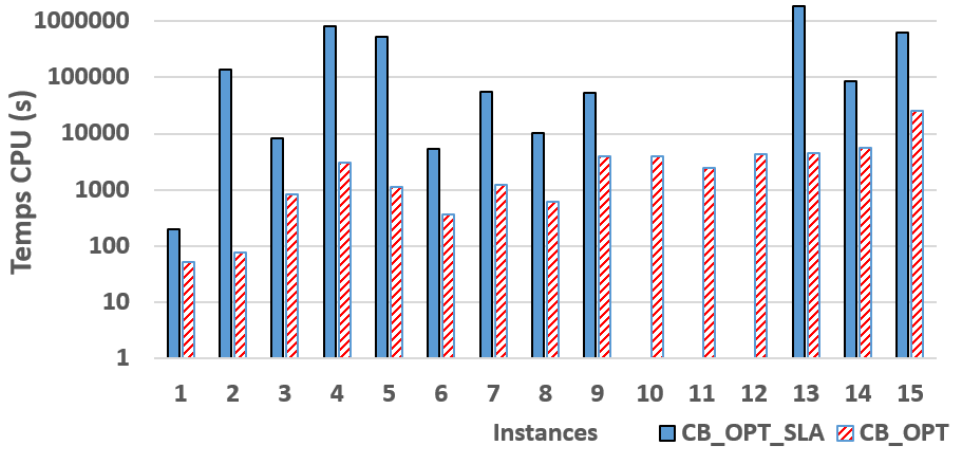


Figure 7.3 Temps CPU (toutes les contraintes)

Le temps d'exécution a également été comparé et les résultats sont illustrés à la Figure 7.3. En général, on observe un comportement semblable aux résultats présentés à la Figure 7.1, où *CB_OPT_SLA* prend plus de temps pour explorer l'espace de recherche, dans le but de déterminer la configuration optimale, exempte de toute violation de contraintes.

7.2 Résultats complémentaires du volet 2

Dans la seconde phase de notre travail, nous avons proposé une approche basée sur les métaheuristiques pour résoudre le modèle d'optimisation présenté au chapitre 4. Toutefois,

des deux scénarios considérés lors de l'implémentation, seuls les résultats du scénario sans contraintes de co-localisation ont été illustrés dans l'article présenté au chapitre 5. Aussi, cette section visera à mettre en évidence le comportement de la métaheuristique proposée, à savoir son aptitude à pouvoir déterminer de "bonnes" solutions en un temps polynomial, dans le cas des instances intégrant les contraintes de co-localisation.

7.2.1 Comparaison avec la méthode exacte

La comparaison s'est d'abord effectuée par rapport à la borne inférieure obtenue avec CPLEX, en considérant les paramètres de simulation déterminés à la section 5.6.1 et les instances présentées au Tableau 5.8, auxquelles nous avons ajouté des contraintes de co-localisation. L'empreinte carbone et le temps d'exécution pour la méthode exacte, *EXACT_CBF* sont présentés dans les colonnes 3 et 4, alors que l'empreinte carbone de la méthode approchée, *ILS_CBF*, son écart par rapport à la valeur optimale, et les temps CPU associés (valeurs minimale, moyenne et maximale) sont illustrés dans les 5 dernières colonnes du Tableau 7.3.

Dans un premier temps, nous pouvons observer que pour les instances où le nombre de serveurs et de VMs sont relativement élevés, soit plus d'1/3 des jeux de données, CPLEX ne parvient pas à déterminer la solution du problème, en raison des limites de ressources ("RL") de la machine utilisée. Dans ce même ordre d'idées, il arrive également que l'exécution s'arrête à l'étape du prétraitement (voir les instances 11, 20, 21, 24, par exemple). Dans ce cas de figure, aucune valeur de CPU ne peut être relevée, comme l'indique le symbole "*". Toutefois, dans le cas des instances qui ont pu être résolues optimalement, les résultats démontrent que les valeurs d'empreinte carbone obtenues avec la méthode approchée *ILS_CBF* sont assez proches de la borne inférieure avec un écart se situant entre 0% et 3.82%. Le signe "-" indique que l'écart n'a pas pu être calculé, en raison de la valeur de borne inférieure non trouvée. En ce qui a trait à la durée d'exécution, nous observons que le temps de calcul de la méthode approchée est non seulement beaucoup plus faible, mais il évolue également de façon linéaire, contrairement à une croissance exponentielle pour la méthode exacte. Cette approche proposée est d'autant plus intéressante à exploiter car elle permet toujours de trouver une solution, contrairement à la méthode exacte qui se révèle totalement inefficace lorsque la taille du problème devient relativement élevée. Ces résultats permettent de mettre en évidence les bonnes performances de l'heuristique proposée, à savoir qu'elle permet d'obtenir un bon équilibre entre la qualité de la solution et le temps d'exécution.

Tableau 7.3 Comparaison entre la méthode exacte *EXACT_CBF* et *ILS_CBF*

Inst.	InterCloud (dimension)	*EXACT_CBF*		*ILS_CBF*				
		Emp. Carb.	CPU	Emp. Carb.	Écart	CPU (s)		
	(D,X,S,V)	(Kg)	(s)	(Kg)	(%)	Min	Moy	Max
1	1, 4, 20, 20	1469.18	9.27	1495.48	1.79	0.55	0.57	0.69
2	1, 4, 40, 40	1562.66	49.31	1580.93	1.17	0.58	0.61	0.67
3	1, 4, 40, 80	1818.55	471.67	1818.55	0.00	0.72	0.76	0.83
4	1, 4, 60, 60	1707.41	5272.77	1707.41	0.00	0.67	0.74	0.86
5	1, 4, 60, 120	2131.16	3292.97	2212.56	3.82	1.53	1.61	1.67
6	1, 8, 40, 40	1562.66	55.48	1580.93	1.17	0.58	0.59	0.61
7	1, 8, 40, 80	1818.55	560.83	1818.55	0.00	0.70	0.81	1.09
8	1, 8, 80, 160	RL	9438.28	2451.32	-	2.20	2.32	2.46
9	1, 8, 120, 120	RL	6667.66	2212.56	-	2.17	2.28	2.45
10	1, 8, 120, 240	RL	9438.28	2781.22	-	7.72	8.11	8.59
11	1, 12, 180, 360	RL	*	3361.00	-	29.23	32.08	33.47
12	2, 8, 40, 20	1090.23	91.95	1090.23	0.00	0.52	0.54	0.55
13	2, 8, 80, 40	1140.01	703.70	1140.01	0.00	0.61	0.62	0.64
14	2, 8, 80, 80	1295.86	12321.63	1295.86	0.00	0.88	0.93	0.97
15	2, 8, 120, 60	1225.59	2900.55	1225.59	0.00	0.81	0.85	0.88
16	2, 8, 120, 120	1504.33	50615.55	1560.52	3.74	2.27	2.42	2.59
17	2, 16, 80, 40	1140.01	1076.64	1140.01	0.00	0.59	0.62	0.64
18	2, 16, 80, 80	1295.86	14953.28	1295.86	0.00	0.84	0.87	0.92
19	2, 16, 160, 80	1295.86	40684.50	1295.86	0.00	1.13	1.17	1.27
20	2, 16, 160, 160	RL	*	1767.68	-	3.33	3.72	4.17
21	2, 16, 240, 120	RL	*	1560.52	-	3.25	3.70	3.91
22	2, 16, 240, 240	RL	*	1995.73	-	15.03	16.27	17.15
23	2, 24, 120, 60	191.98	3826.92	191.98	0.00	0.77	0.81	0.84
24	2, 24, 240, 120	RL	*	246.18	-	3.09	3.42	3.73
25	2, 24, 240, 240	RL	*	317.10	-	13.37	14.68	16.02
26	2, 24, 360 180	RL	*	268.44	-	9.15	9.86	10.28
27	2, 24, 360, 360	RL	*	406.76	-	53.42	58.27	62.42
28	3, 12, 60, 20	895.32	939.14	895.32	0.00	0.53	0.56	0.59
29	3, 12, 60, 40	971.02	5460.78	971.02	0.00	0.59	0.61	0.63
30	3, 12, 120, 40	971.02	9683.16	971.02	0.00	0.67	0.70	0.74

7.2.2 Comparaison avec des méthodes de référence

Dans une seconde phase, nous avons également comparé l'approche de résolution proposée avec des méthodes de référence, et l'ensemble des résultats relatifs au coût d'empreinte carbone et au temps d'exécution de chaque algorithme sont illustrés au Tableau 7.4.

De manière générale, on observe des résultats plus ou moins identiques en termes de comportements des différents algorithmes en ce qui a trait aux scénarios avec et sans contraintes. En effet, l'heuristique proposée permet en général d'obtenir la configuration de coût minimal, car elle tente d'améliorer la solution initiale en explorant efficacement l'espace de recherche. Pour ce qui du temps d'exécution, on dénote que la méthode proposée est, de loin, beaucoup moins rapide que les autres approches de référence. Ceci s'explique par le fait que les

Tableau 7.4 Comparaison entre *ILS_CBF* et des algorithmes de référence

Inst.	InterCloud (dimension)	*ILS_CBF*			*GDCF_I*		*FF_A*		*BFD_A*	
		Emp. Carb.	CPU	CPU-sol	Emp. Carb.	CPU	Emp. Carb.	CPU	Emp. Carb.	CPU
	(D,X,S,V)	(t)	(s)	(s)	(t)	(s)	(t)	(s)	(t)	(s)
1	1, 12, 60, 60	1.61	0.69	0.01	1.63	0.01	1.64	0.00	1.63	0.01
2	1, 12, 120, 120	2.05	2.35	0.02	2.10	0.06	2.07	0.01	2.10	0.10
3	1, 12, 120, 240	2.55	7.99	0.04	2.66	0.22	2.57	0.01	2.66	0.33
4	1,12, 180, 180	2.33	6.83	0.04	2.42	0.22	2.36	0.00	2.42	0.29
5	1, 12, 180, 360	3.36	35.61	0.10	3.58	0.93	3.37	0.02	3.58	0.98
6	3, 12, 60, 40	1.00	0.62	0.01	1.00	0.01	1.17	0.00	1.00	0.01
7	3, 24, 120, 80	1.24	1.28	0.01	1.24	0.02	1.30	0.00	1.24	0.05
8	3, 30, 300, 1800	26.85	2795.79	2744.69	28.21	6.88	27.80	0.53	28.21	25.41
9	3, 30, 300, 2100	39.46	4432.63	3888.70	48.75	10.86	47.83	1.07	48.75	31.71
10	4, 40, 200, 1000	6.29	280.15	0.80	6.29	1.52	6.29	0.12	6.29	7.56
11	4, 40, 400, 2000	8.57	3684.74	5.37	8.57	12.62	8.57	0.49	8.57	51.07
12	6 ,60, 600, 3000	14.52	14017.00	17.47	14.52	19.55	14.52	1.09	14.52	144.60
13	6, 120, 600, 3000	18.32	7028.26	17.50	18.32	17.02	18.32	1.07	18.32	135.27
14	8, 160, 800, 4000	22.81	19230.20	120.16	23.01	31.37	22.86	2.87	23.01	683.10
15	10, 100, 500, 2500	14.82	4019.04	2078.21	14.39	6.76	14.88	0.68	14.39	87.65

autres méthodes sont des algorithmes gloutons qui s'arrêtent une fois que toutes les VMs sont placées, sans nécessairement chercher à raffiner la solution. En ce sens, afin d'effectuer une comparaison plus équitable, nous avons également présenté le temps au bout duquel la meilleure solution a été retrouvée par notre algorithme (CPU-sol). En analysant ces valeurs, on constate qu'elles se rapprochent beaucoup plus du temps CPU des autres algorithmes gloutons. Ceci nous permet donc de conclure que, bien que le temps d'exécution total soit très élevé, la méthode proposée est en mesure de trouver la meilleure configuration en un temps réduit. Néanmoins, il est important de souligner le temps CPU-sol très élevé observé pour les instances 8, 9 et 15. Ceci peut s'expliquer par le fait que ces espaces de recherche sont beaucoup plus difficiles à explorer en considérant les valeurs clés déterminées à la phase de paramétrisation.

7.3 Résultats complémentaires du volet 3

Le dernier volet de notre travail visait à proposer un modèle d'optimisation global de l'empreinte carbone d'un environnement InterCloud, en y intégrant une modélisation plus complète de la charge à héberger. En effet, le modèle présenté au chapitre 4 a été étendu afin d'inclure la prise en charge des applications complexes, et plus précisément le trafic entre les VMs, le routage de ce dernier et les contraintes s'y rapportant. Comme le modèle d'optimisation résultant est NP-difficile, une approche basée sur une hybridation de deux métaheuristiques a également été proposée afin de solutionner les instances de grande taille. Ainsi,

des simulations visant à mettre en évidence autant les performances du nouveau modèle que celles de la méthode de résolution ont été réalisées. À l'instar du second volet, dans l'article présenté au chapitre 6, nous avons seulement illustré les résultats se rapportant au premier scénario. Aussi, nous nous donnons pour tâche, dans la suite de cette section, de présenter les résultats intégrant les contraintes de co-localisation.

Tableau 7.5 Comparaison de coûts d'empreinte carbone

Inst.	InterCloud (D, T, X, S, V, F)	$CB_{G}\_OPT\_SLA$	VM_{OPT}	NET_{OPT}	EFF_{OPT}
		Emp. Carb. (t)	Emp. Carb. (t)	Emp. Carb. (t)	Emp. Carb. (t)
1	1, 9, 4, 8, 20, 46	4.02	5.47	7.23	8.62
2	1, 9, 4, 8, 30, 106	4.51	5.66	7.61	8.82
3	1, 9, 4, 8, 40, 190	5.04	6.30	8.13	9.35
4	2, 18, 8, 16, 20, 46	2.02	4.48	10.02	6.34
5	2, 18, 8, 16, 30, 106	2.30	4.61	10.08	6.47
6	2, 18, 8, 16, 40, 190	2.70	5.06	10.38	6.83
7	3, 27, 12, 24, 20, 46	1.15	4.18	11.18	4.20
8	3, 27, 12, 24, 30, 106	1.29	4.27	11.43	5.22
9	3, 27, 12, 24, 40, 190	0.00	0.00	11.86	5.41

7.3.1 Comparaison avec des modèles de référence

La première étape des simulations vise à comparer le modèle d'optimisation proposée avec d'autres modèles de référence, comme à la section 6.6.2 du chapitre 6, mais en intégrant les contraintes de co-localisation. Le modèle proposé, $CB_{G}\_OPT\_SLA$ (A), est comparé à une approche optimisant l'empreinte carbone des nœuds de calcul, VM_{OPT} (B), à un modèle qui minimise l'impact de la consommation des ressources réseau, NET_{OPT} (C), et enfin à une approche optimisant l'efficacité énergétique, EFF_{OPT} (D). L'empreinte carbone totale résultant de l'optimisation de chaque modèle est présentée dans le Tableau 7.5, l'écart (en pourcentage) entre le coût retourné par le modèle proposé et celui de chacun des autres modèles est illustré à la Figure 7.4 et la durée d'exécution pour chacune des approches est présentée à la Figure 7.5.

De manière générale, le comportement global du modèle avec et sans contraintes de co-localisation est identique. En effet, le modèle proposé, $CB_{G}\_OPT\_SLA$, optimise conjointement le placement des VMs, le routage du trafic et l'efficacité du data center, de telle sorte que l'énergie totale consommée et l'empreinte carbone associée sont minimales. La Figure 7.4 permet de mieux apprécier les gains d'empreinte carbone obtenus avec $CB_{G}\_OPT\_SLA$, où il est possible d'observer des écarts pouvant s'élever jusqu'à environ 900%. Toutefois, en raison de sa grande complexité, $CB_{G}\_OPT\_SLA$ demeure, en général, l'approche la moins

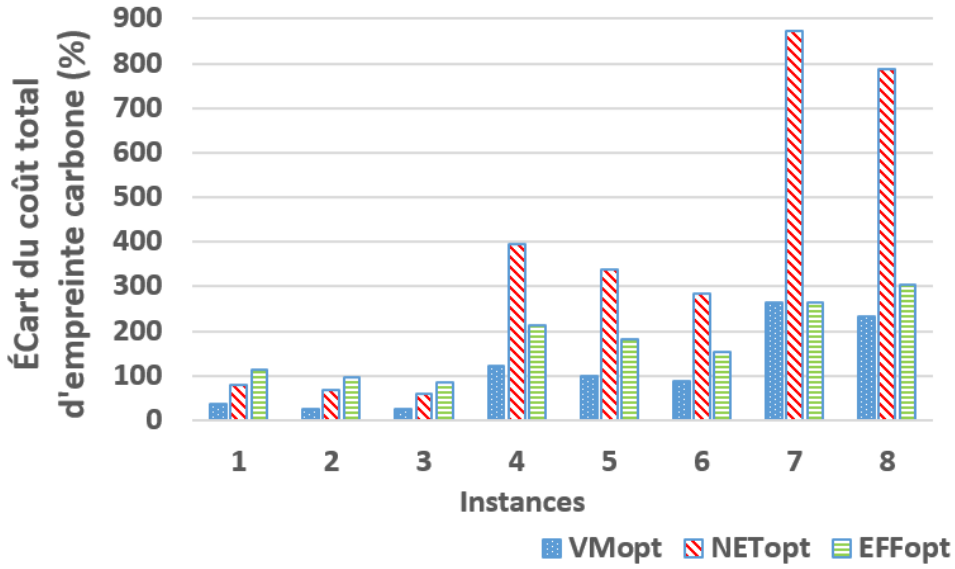


Figure 7.4 Pourcentage d'écart du coût total d'empreinte carbone

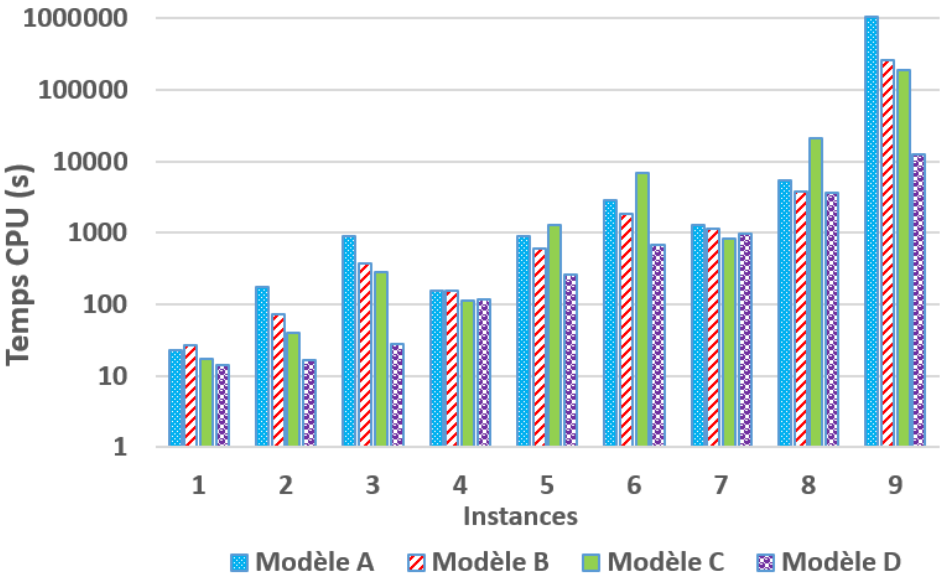


Figure 7.5 Comparaison des temps d'exécution

rapide, comme l'illustre la Figure 7.5.

7.3.2 Comparaison avec la méthode exacte

Afin d'évaluer les performances de l'approche de résolution proposée, nous avons repris les valeurs des paramètres de simulations considérées au chapitre 6 et les instances utilisées au

Tableau 7.6 Comparaison entre la méthode exacte OPT_G_{CBF} et ITS_G_{CBF}

Inst.	InterCloud (D, T, X, S, V, F)	OPT_G_{CBF}	ITS_G_{CBF}					
		Emp. Carb. ($KgCO_2$)	Emp. Carb. ($KgCO_2$)	Min (%)	Écart Moy (%)	Max (%)	Iter. Total (#)	Meilleure Iter. (#)
1	1, 11, 6, 18, 60, 34	4270.75	4270.75	0.00	0.00	0.00	18490	817
2	1, 11, 6, 30, 60, 34	3227.42	3227.42	0.00	0.00	0.00	21595	492
3	1, 11, 6, 30, 76, 31	5043.47	5050.76	0.00	0.14	1.45	24504	1101
4	1, 15, 10, 30, 60, 34	5050.87	5107.04	0.26	1.11	2.42	25658	6840
5	1, 15, 10, 50, 60, 34	3217.98	3297.38	2.47	2.47	2.47	40570	403
6	1, 9, 4, 12, 56, 24	11131.66	11172.14	0.34	0.36	0.40	38580	16612
7	1, 9, 4, 16, 56, 24	9377.61	10385.73	9.91	10.75	11.67	50086	35861
8	2, 22, 12, 60, 60, 34	2972.66	3030.81	1.96	1.96	1.96	41742	10
9	2, 30, 20, 100, 60, 34	2740.07	2810.46	0.78	2.57	6.57	53989	11274
10	2, 18, 8, 24, 60, 30	12328.54	12355.22	0.11	0.22	0.30	26687	6412
11	2, 22, 12, 36, 46, 32	2937.75	3464.57	1.68	17.93	19.74	20293	1267
12	2, 22, 12, 36, 52, 32	3050.41	3087.07	0.50	1.20	3.73	24710	9063
13	2, 22, 12, 60, 46, 32	2742.17	2785.63	0.67	1.58	2.12	28174	203
14	2, 22, 12, 60, 52, 32	2917.44	2952.33	0.00	1.20	1.99	32014	32
15	3, 33, 18, 54, 60, 34	2566.64	2613.54	0.00	1.83	3.35	37751	8083
16	3, 33, 18, 54, 46, 32	2437.97	2437.97	0.00	0.00	0.00	26924	19
17	3, 33, 18, 54, 52, 32	2641.56	2655.14	0.10	0.51	1.35	41576	12682
18	3, 33, 18, 90, 46, 32	2437.97	2437.97	0.00	0.00	0.00	34717	24
19	3, 33, 18, 90, 52, 32	2641.56	2644.30	0.10	0.10	0.10	39577	39
20	4, 44, 24, 72, 51, 45	1953.49	2028.90	1.13	3.86	6.29	43897	13901

Tableau 6.3, en y intégrant les contraintes de co-localisation. Par la suite, l'algorithme de résolution exacte, OPT_G_{CBF}, et la méthode proposée, ITS_G_{CBF}, ont été utilisés pour résoudre les différentes tailles de problème et les coûts d'empreinte carbone ont été analysés. Plus particulièrement, le Tableau 7.6 présente l'empreinte carbone obtenue avec les deux méthodes, de même que le pourcentage d'écart (minimal, moyen et maximal) de la méthode approchée, le nombre d'itérations totales ainsi que l'itération à laquelle la meilleure solution a été trouvée par ITS_G_{CBF}.

Le Tableau 7.6 démontre que les valeurs d'empreinte carbone obtenues avec la méthode proposée sont généralement assez proches de la borne inférieure : l'algorithme est en mesure de trouver la solution optimale pour les instances 1, 2, 3, 14, 15, 16 et 18, et affiche, en moyenne, un pourcentage d'écart moyen inférieur à 3%. Toutefois, nous observons également que la distance par rapport à la borne inférieure peut s'élever jusqu'à environ 20% dans le cas de l'instance 11, comme pour le scénario sans contraintes. Ceci est dû au fait que pour cette instance en particulier, et les paramètres de simulations déterminés, les sauts dans le voisinage ne permettent pas d'explorer efficacement un tel espace de recherche et l'algorithme demeure piégé dans un minimum local. Toutefois, le choix des paramètres de

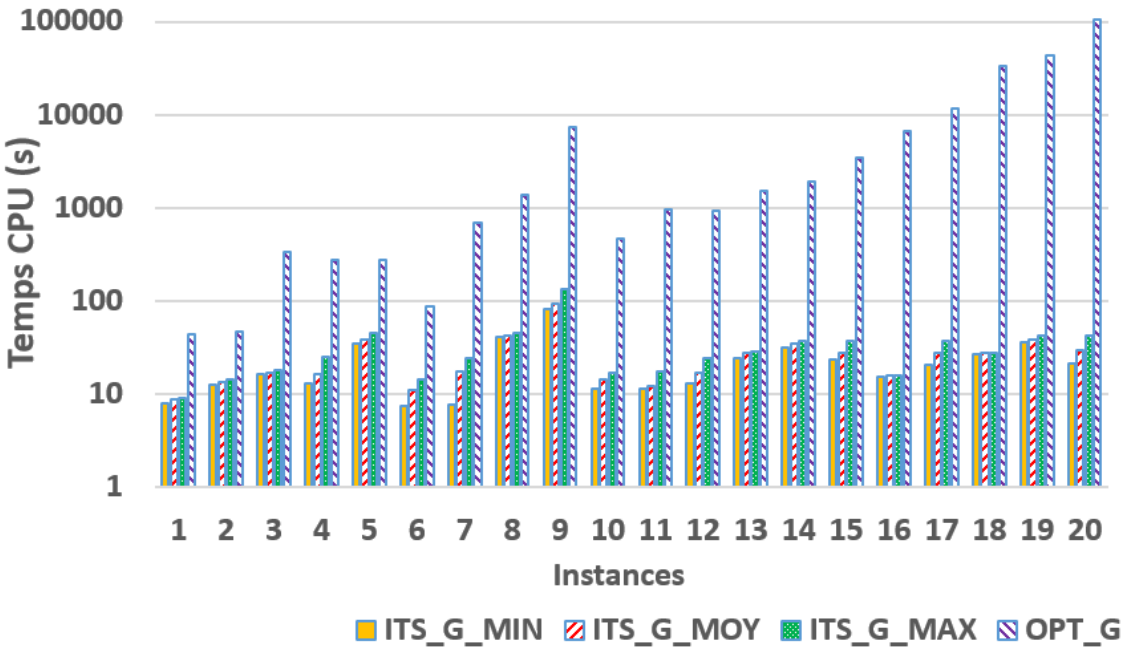


Figure 7.6 Comparaison du temps d'exécution entre ITS_G_{CBF} et OPT_G_{CBF}

simulations demeure pertinent car l'algorithme est en mesure de solutionner, en général, différentes tailles de problème, où la solution sous-optimale peut être trouvée autant au début de la phase d'exploration, comme pour les instances 8, 14, et 16, par exemple, ou au bout de plusieurs itérations, dans le cas des espaces de recherche beaucoup plus difficiles à explorer.

Les performances de l'algorithme ITS_G_{CBF}, en termes de temps d'exécution ont également été considérées et sont illustrées à la Figure 7.6. Cette dernière présente les valeurs minimale, moyenne et maximale de la durée d'exécution de ITS_G_{CBF}, de même que le temps de calcul pris par la méthode exacte. Les résultats démontrent que le temps CPU de l'algorithme proposé croît linéairement par rapport à la taille du problème, avec des temps de calcul se situant entre 7.24 et 135 secondes, alors que la méthode exacte peut prendre jusqu'à environ 30 heures (105286 secondes) pour résoudre un problème.

Ainsi, avec un écart maximal moyen d'environ 3.3% et un temps d'exécution moyen inférieur à 100 secondes, les résultats ont su clairement mettre en évidence les bonnes performances de ITS_G_{CBF} par rapport à la qualité des solutions et le temps d'exécution.

Tableau 7.7 Comparaison entre ITS_G_{CBF} et des méthodes de référence

Inst.	InterCloud (D, T, X, S, V, F)	ITS_G_{CBF}			*RAND*	*ILS*	*TS*	*ITS_RAND*
		Emp. Carb. (t)	CPU (s)	CPU-sol (s)	Emp. Carb. (t)	Emp. Carb. (t)	Emp. Carb. (t)	Emp. Carb. (t)
1	1, 15, 10, 50, 430, 931	26.71	21.30	13.57	997093.10	26.90	26.74	29.29
2	1, 15, 10, 50, 550, 1151	33.04	47.67	40.83	3113779.00	33.11	33.11	34.45
3	2, 30, 20, 100, 500, 929	17.34	60.99	39.03	89.00	17.57	17.57	22.64
4	2, 30, 20, 100, 500, 1021	15.45	61.71	46.80	46.70	15.52	15.50	22.55
5	4, 60, 40, 200, 580, 805	11.63	66.10	11.98	80.52	11.64	11.64	22.07
6	4, 60, 40, 200, 580, 1057	12.13	159.96	33.11	61.81	12.13	12.13	22.06
7	6, 90, 60, 300, 560, 817	9.33	201.92	32.13	109.14	9.33	9.33	31.44
8	6, 90, 60, 300, 770, 1189	15.50	475.52	124.52	142.32	15.55	15.55	38.86
9	8, 120, 80, 400, 660, 805	13.05	332.99	24.85	176.63	13.06	13.06	46.49
10	8, 120, 80, 400, 770, 931	14.39	680.90	88.85	209.73	14.42	14.42	48.99
11	10, 150, 100, 500, 860, 1026	16.61	893.89	217.87	226.80	16.67	16.67	55.63
12	10, 150, 100, 500, 890, 1007	14.68	788.66	44.89	252.22	14.68	14.68	56.13

7.3.3 Comparaison avec des méthodes de référence

Afin d'évaluer l'efficacité générale de ITS_G_{CBF}, nous avons également comparé les performances de l'algorithme à celles d'autres méthodes approchées, comme à la section 6.6.3.2, en considérant différentes tailles de problème. Les coûts d'empreinte carbone résultant de l'exécution de ITS_G_{CBF}, de même que le temps de calcul total et le temps auquel la meilleure solution a été trouvée (CPU-sol) sont présentés dans les colonnes 3 à 5 du Tableau 7.7. Les valeurs d'empreinte carbone obtenues avec les méthodes de référence y sont également illustrées. De manière générale, les résultats du Tableau 7.7 démontrent que la méthode proposée permet toujours d'obtenir la configuration de coût minimal. Nous pouvons également observer que l'écart en termes de coût d'empreinte carbone est particulièrement importante dans le cas de l'algorithme *RAND*, car ce dernier détermine, de manière aléatoire, l'emplacement de chaque VMs sans aucune considération pour les différentes contraintes associées au problème, ce qui peut être à l'origine de configurations non faisables. Aussi, bien que l'écart entre la méthode proposée et ses deux premières variantes, *ILS* et *TS* sont faibles, ITS_G_{CBF} demeure le meilleur choix car il permet de conjointement tirer profit des avantages de chacun de ces algorithmes. Toutefois, bien que la dernière variante de ITS_G_{CBF}, *ITS_RAND*, implémente également ces mécanismes, cet algorithme ne performe pas aussi bien car, avec une solution initiale générée aléatoirement, l'exploration peut s'avérer difficile, particulièrement si l'heuristique reste piégée dans un optimum local de piètre qualité. La différence est encore plus évidente à la Figure 7.7 qui illustre le meilleur coût d'empreinte carbone obtenu avec les 2 méthodes. De manière générale, nous observons que l'écart augmente en fonction de la

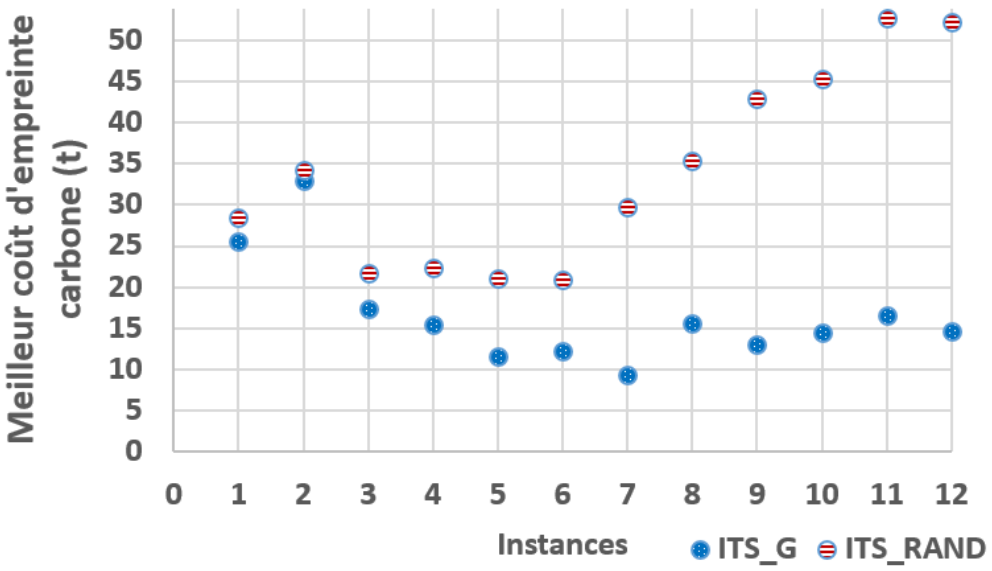


Figure 7.7 Comparaison des meilleurs coûts d'empreinte carbone entre ITS_G_{CBF} et ITS_RAND

taille du problème, ce qui souligne le côté évolutif de l'approche de résolution proposée.

Ces deux algorithmes ont été également comparés en termes de temps d'exécution. Les valeurs minimale, moyenne et maximale du temps CPU total pris par chaque méthode ont été illustrées à la Figure 7.8, alors que le meilleur temps auquel la solution sous-optimale a été trouvée par chaque algorithme est présenté à la Figure 7.9. De la Figure 7.8, nous pouvons observer que les performances de ITS_G_{CBF} dépassent largement celles de la méthode comparative en termes de temps d'exécution également. De plus la méthode proposée est d'autant plus efficace car, non seulement la meilleure solution retournée par ITS_RAND est de mauvaise qualité (voir la Figure 7.7), mais le temps nécessaire pour l'obtenir est, en général, supérieur à celui de l'algorithme développé, comme l'illustre la Figure 7.9.

Dans l'ensemble, ces résultats ont permis de démontrer l'importance d'utiliser l'approche de résolution proposée, ITS_G_{CBF}, afin de résoudre le problème de placement des applications. En effet, l'algorithme fait preuve d'une grande flexibilité en tirant avantages des caractéristiques combinés des métaheuristiques ILS et TS, ce qui lui permet d'obtenir de "bonnes" solutions en un temps polynomial.

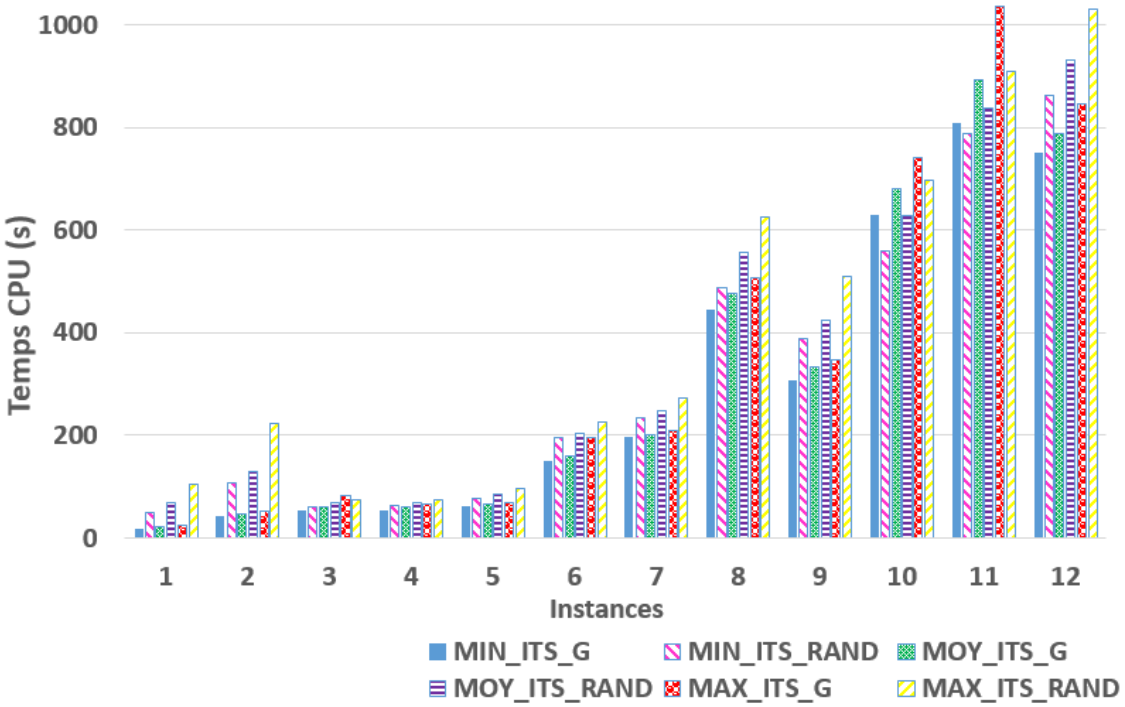


Figure 7.8 Comparaison des temps d'exécution entre ITS_G_{CBF} et ITS_RAND

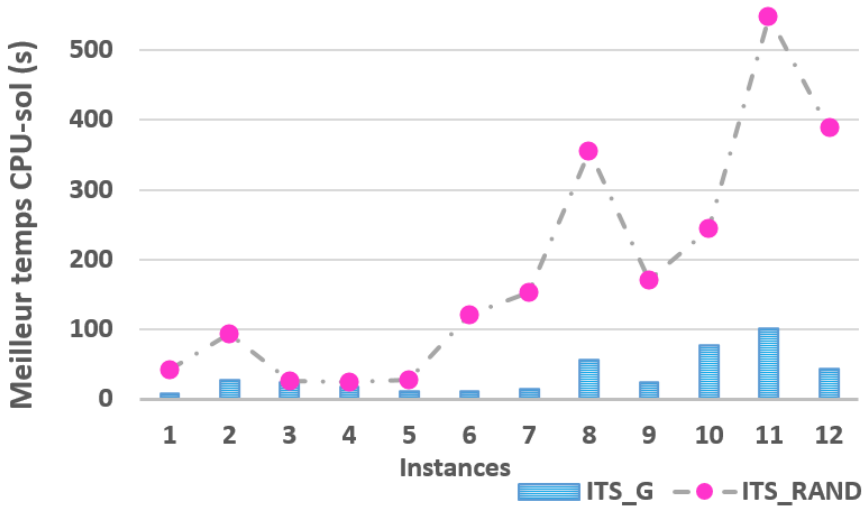


Figure 7.9 Comparaison des meilleurs temps CPU-sol entre ITS_G_{CBF} et ITS_RAND

7.4 Analyse sommaire des résultats

Cette section visait à présenter les résultats ne pouvant être illustrés aux chapitres 4 à 6, particulièrement dans le but de mieux renforcer la mise en évidence des contributions des travaux réalisés.

En effet, grâce aux résultats de la section 7.1, nous avons pu souligner l'efficacité du modèle d'optimisation proposé, $CB_G\_OPT\_SLA$, en ce qui a trait au respect des différentes contraintes pouvant survenir lors de la consolidation de charges de différents types. Toutefois, en raison de la grande complexité du modèle développé, le temps de calcul évoluait de manière exponentielle.

Aussi, dans le but de déterminer des solutions sous-optimales en un temps réduit, nous avons également développé, aux chapitres 5 et 6, des méthodes de résolution approchées visant à solutionner le problème de placement des charges. Toutefois, comme à la section 7.1, seul un nombre très restreint d'instances de petite taille, intégrant l'ensemble des contraintes du problème, ont pu être résolues de manière exacte avec CPLEX, seules les contraintes de co-localisation ont été considérées lors de l'implémentation des algorithmes de placement. Pour compléter les résultats présentés aux chapitres 5 et 6, nous avons alors également analysé les performances des méthodes de résolution lorsque les contraintes de co-localisation sont considérées. De manière générale, pour les instances et les paramètres de simulations considérés aux chapitres 5 et 6, nous avons observé des comportements similaires dans les 2 scénarios (avec et sans contraintes), à savoir que ces algorithmes permettent d'obtenir de "bonnes" solutions en un temps réduit, ce qui permet de souligner en général, les performances des méthodes de résolution proposées et la justesse au niveau du choix des paramètres de simulation qui y sont associés.

En somme, une fois l'ensemble des travaux présentés, vient alors le temps de jeter un regard critique sur la démarche méthodologique qui a été utilisée afin d'obtenir ces résultats. Tel est l'objectif visé dans le chapitre suivant.

CHAPITRE 8 DISCUSSION GÉNÉRALE

Dans ce chapitre, nous tenterons de porter un certain jugement sur l'ensemble des travaux réalisés dans le cadre de cette thèse. Plus particulièrement, il sera d'abord question d'effectuer une analyse en regard des aspects méthodologiques en vue d'atteindre les objectifs fixés. Par la suite, une seconde étape visera à porter un oeil critique sur l'ensemble des résultats découlant de nos travaux afin de mieux dégager la portée de ces derniers.

8.1 Analyse méthodologique

L'objectif principal de cette thèse consistait à mettre en place un cadre de planification des applications dans un environnement InterCloud afin de réduire l'empreinte carbone de ce dernier. Afin de réaliser cette tâche, nous avons subdivisé le travail en trois parties distinctes.

Dans le premier volet de la thèse, nous avons développé un modèle d'optimisation de l'empreinte carbone pour les applications à VM unique. Pour rendre le modèle le plus vraisemblable possible, plusieurs aspects se rapportant à l'hétérogénéité de l'infrastructure et des demandes, aux exigences des utilisateurs et aux contraintes de performances ont été considérés. Une résolution manuelle, sur de petites instances du problème, nous a permis de valider la prise en compte de ces différents aspects. Par la suite, le modèle a été résolu à l'aide de l'outil de programmation mathématique, AMPL, et du solveur CPLEX. Vu qu'aucun travail similaire au nôtre n'avait été répertorié dans la littérature, comparer nos résultats avec ceux d'autres modèles n'était pas envisageable. Ainsi, afin d'évaluer notre modèle, nous avons eu à adapter certains modèles existants ou encore à en concevoir de nouveaux, dont les buts poursuivis étaient sensiblement similaires à nos objectifs. Plusieurs simulations ont mis en évidence la pertinence et l'utilité du modèle proposé.

Aussi, puisque ce genre de problème est difficile à résoudre, particulièrement pour les grandes instances, nous avons, dans le second volet de notre travail, conçu une méthode de résolution à base de métaheuristique afin de trouver des solutions sous-optimales, mais en un temps polynomial. Les performances de l'heuristique ont été comparées aux solutions obtenues avec le modèle exact, mais aussi à des méthodes de résolution connues, mais que nous avons adaptées à notre problème, à des fins de comparaison équitable.

Dans le dernier volet de la thèse, nous avons développé un modèle de placement plus général, où des applications plus complexes et les contraintes associées ont été prises en compte. Ce modèle complet est encore plus difficile à résoudre. De ce fait, nous avons également conçu

un algorithme hybride afin de solutionner ce problème d'optimisation.

À ce stade, il est de notre devoir de porter un regard critique sur la démarche utilisée et de discuter des choix que nous avons effectués en regard des alternatives qui s'offraient à nous. À cet effet, cette analyse sera divisée en deux parties, la première portant sur le modèle d'optimisation développé, et la seconde, sur les méthodes de résolution et les outils utilisés.

8.1.1 Modèle d'optimisation

Bien que plusieurs travaux aient porté sur l'optimisation de la puissance consommée ou l'empreinte carbone d'un environnement Cloud, l'ensemble des modèles développés demeurent inappropriés, car ils ne considèrent pas, à la fois, tous les paramètres facilitant une évaluation plus précise de la consommation énergétique et l'empreinte carbone associée. Ainsi, le but principal de cette thèse était de proposer un nouveau modèle d'optimisation de l'empreinte carbone qui, non seulement intègrerait tous les facteurs clés influençant l'énergie et l'impact écologique d'un InterCloud, tels les serveurs et leurs ventilateurs, les ressources réseau et le cooling, mais qui permettrait également d'étudier les relations complexes entre ces différentes entités, tout en considérant divers types de requis imposés par les utilisateurs.

8.1.1.1 Placement statique des applications

En vue de mettre en évidence l'apport d'un tel modèle, il était plus convenable de considérer un scénario simple de placement statique des applications, comme il a été fait dans de nombreux travaux antérieurs. En fait, le cadre considéré dans notre travail peut se résumer à une situation où un fournisseur d'infrastructure Cloud, propriétaire de plusieurs data centers initialement vides, reçoit un certain nombre d'applications simples ou complexes à héberger pour une durée bien définie. Vu que l'objectif premier de cette thèse était d'évaluer l'impact des interactions simultanées entre la consommation dynamique des serveurs, le routage du trafic inter-VMs et l'efficacité du système de refroidissement, sous la contrainte d'exigences de Qualité de Service (QdS), nous avons préféré opter pour un cadre de planification nous permettant de développer un modèle plus exact et ne faisant intervenir que ces différents éléments dans l'évaluation de l'empreinte carbone d'un InterCloud. Ceci a été fait afin d'assurer que l'effet de ces interactions et de cette optimisation conjointe sur la réduction de l'impact écologique d'un InterCloud soit clairement et convenablement illustré. En ce sens, par souci de simplicité, nous avons considéré un contexte de placement exempt de tout mécanisme de migration des applications. Comme ce travail, au meilleur de nos connaissances, est la toute première tentative consistant à intégrer ces relations complexes dans le processus de placement des applications et de la minimisation de l'empreinte écologique d'un InterCloud,

introduire le processus de migration des charges risquerait d'entraver la mise en évidence d'une telle contribution.

C'est dans ce même ordre d'idées que nous avons supposé que la charge à héberger, en termes de nombres de VMs et de trafics associés, est connue à l'avance, tout comme leurs besoins en ressources physiques qui demeurent fixes tout au long de la durée de l'hébergement. Cette invariabilité des besoins en ressources physiques est une conséquence directe de l'absence de prise en compte de la migration, car si les applications devenaient gourmandes en ressources, il faudrait implémenter des mécanismes de migration permettant de relocaliser ces VMs afin de continuer à satisfaire la demande. Aussi, pour ce qui est du trafic, sans perte de généralité, nous avons uniquement considéré les interactions VMs-VMs, car au même titre que les trafics VMs-utilisateurs, nous avions eu à gérer la proximité et le délai, le routage du trafic et la consommation des équipements réseau. En ce sens, bien que le modèle proposé ne considère qu'un seul type de trafic, il peut aisément être adapté afin d'intégrer les interactions VMs-usagers, ces derniers n'introduisant aucun défi supplémentaire lié à l'hébergement des trafics.

8.1.1.2 Exigences des clients

Afin de rendre le modèle de planification plus réaliste, nous avons considéré deux types de contraintes de QdS : le premier, lié à la nature intrinsèque des applications et dont la violation peut entraîner une dégradation des performances des applications, et le second reflétant plus les exigences des clients en matière de sécurité, de territorialité et de redondance. Concernant la dégradation de performance, plusieurs approches ont été présentées dans la littérature. Toutefois, la plupart utilise un modèle de performance basée sur la variabilité des charges qui, afin de respecter des contraintes de performances, induirait, à l'occasion, des processus de migration. Or, comme ce dernier aspect n'est pas considéré dans notre approche, nous avons préféré adapter le modèle proposé par Sharifi *et al.* (2012) où les performances sont directement liées à la nature des applications co-hébergées.

Pour ce qui est des exigences relatives aux contraintes de co-localisation, nous n'avons pas pu recenser de travaux traitant de ces aspects. De ce fait, afin d'intégrer ces exigences du client à notre approche, nous avons supposé l'existence de matrices traduisant les interférences entre les différentes VMs à héberger. Plus particulièrement, lors de la soumission de nouvelles applications, chaque client définit ses besoins en termes de VMs et des contraintes de localisation associées. Des matrices d'interférence sont alors dérivées, en considérant ces contraintes. Vu que l'un des objectifs spécifiques de la thèse visait à modéliser, par le biais d'équations mathématiques, les contraintes de co-hébergement des VMs, afin de considérer les requis en termes de sécurité et de redondance imposés par le client, par souci de simplicité,

nous avons supposé qu'une étape antérieure au processus de placement permet de traduire les exigences des clients en matrices d'interférence, lesquelles sont par la suite utilisées comme paramètres d'entrées à notre modèle d'optimisation.

Dans notre modélisation des applications complexes, nous avons également imposé que toutes les VMs d'une application donnée soient hébergées dans le même data center. Ce choix de modélisation des contraintes peut s'expliquer de deux façons. Dans un premier temps, cette contrainte permet d'enrayer tout délai de communication inter-VMs, le délai à l'intérieur d'un data center étant négligeable. Dans un second temps, bien que les data centers appartiennent à un même fournisseur Cloud, nous supposons qu'ils sont interconnectés par un réseau dorsal, contrôlé par une autre entité. De ce fait, en l'absence d'une telle contrainte, il serait possible d'héberger des VMs d'une même application dans différents data centers, ce qui introduirait du trafic inter data centers non négligeables. Or, comme le réseau dorsal n'est pas géré, selon le cadre défini dans notre modèle, par le fournisseur Cloud, il serait impossible, du point de vue de ce dernier, de déterminer le mécanisme de routage efficace du trafic à travers ce réseau d'interconnexion. De même, le surplus d'empreinte carbone associée aux commutateurs du réseau dorsal d'interconnexion ne saurait être imputable au fournisseur d'infrastructure. Telles sont les principales raisons qui nous ont portés à limiter le trafic inter-VMs à l'intérieur des data centers.

8.1.1.3 Infrastructure physique

Nous avons également eu à effectuer des choix se rapportant à l'infrastructure physique. Dans notre modélisation de la distribution de la température à l'intérieur des data centers, nous avons fait abstraction du phénomène de recirculation de chaleur afin de clairement mettre en évidence les interactions entre la nature dynamique des ventilateurs des serveurs et l'efficacité du système de refroidissement, en fonction de la température fournie par ce dernier. En considérant le phénomène de recirculation, cette mise en évidence serait altérée, car la température d'entrée des châssis serait le résultat de la température fournie par le système de cooling, combinée à la température de l'air chaud recirculant à l'intérieur du data center. Il est également important de noter que cette hypothèse d'une absence de recirculation de chaleur n'est pas dénuée de sens. En effet, en raison de sa faible densité, l'air chaud s'échappant des serveurs a tendance à monter vers le plafond, pour être ensuite aspiré par des bouches d'évacuation. Toutefois, il peut arriver qu'un faible pourcentage de cet air chaud ne soit pas aspiré et retourne dans la salle pour se combiner à l'air froid pénétrant l'entrée des serveurs. En supposant la présence de séparateurs physiques qui délimitent les rangées d'air chaud des rangées d'air froid, il est possible d'éliminer ce phénomène de recirculation. Pour ce qui

est du schéma d'interconnexion des serveurs à l'intérieur des data centers, notre choix s'est arrêté sur la topologie en arbre. Ce choix découle particulièrement du fait que, d'une part, c'est la topologie la plus couramment utilisée, et d'autre part, parce que les autres topologies, plus complexes à implémenter, présentent certains avantages qui ne sont pas nécessairement pertinents à exploiter en regard des objectifs visés par cette thèse.

8.1.2 Outils et méthodes de résolution

Une fois le modèle d'optimisation proposé, plusieurs approches de résolution s'offraient à nous. Dans un premier temps, nous avons opté pour une méthode de résolution exacte afin de mettre en évidence la contribution de notre modèle d'optimisation par rapport aux modèles existants, et par la suite, nous avons également développé des méthodes approchées afin de résoudre les problèmes de grande taille.

8.1.2.1 Méthode de résolution exacte

Le modèle d'optimisation développé étant de nature non linéaire, utiliser un solveur mathématique traitant des problèmes de ce type serait l'option naturelle à choisir. Toutefois, ces problèmes étant complexes à résoudre, des expériences préliminaires nous ont démontré que l'utilisation de solveurs non linéaires entraînait des temps de calcul excessivement élevés. De ce fait, afin de contourner le problème, nous avons transformé le modèle en un système totalement linéaire. Comme la non-linéarité du modèle initial était due à la variable de température, nous avons usé d'une astuce d'implémentation qui, pour un ensemble de data centers, fait varier la température de ces dernières, et pour chaque combinaison de valeurs de température fixée, la version linéaire du modèle est résolue à l'aide du solveur CPLEX. La combinaison de valeurs de température entraînant l'empreinte carbone la plus faible est alors retenue comme solution optimale. Aussi, en dépit du fait qu'il existe plusieurs solveurs linéaires pour la résolution exacte, notre choix s'est arrêté sur CPLEX en raison de la puissance du logiciel et des options de réglages disponibles nous permettant d'améliorer les processus d'exploration et de branchement du solveur. Bien que nous ayons utilisé les réglages de base du logiciel, ce dernier a pu démontrer d'excellentes performances dans la recherche de la solution optimale. Un ajustement des réglages permettrait sans doute d'optimiser les mécanismes et l'utilisation de la mémoire interne, ce qui faciliterait, sans doute, la résolution d'instances de taille beaucoup plus élevée. Une autre question relative à la méthodologie porte sur le niveau de granularité à utiliser lorsqu'il faut faire varier la température dans l'intervalle d'intérêt. Dans notre implémentation, nous avons considéré des valeurs entières de température et donc une précision à l'unité. Une granularité plus fine permettrait d'obtenir des valeurs plus précises, toutefois,

ce serait au terme d'un plus long temps d'exécution, dû aux nombreuses combinaisons de valeurs de température à tester. De plus, ceci ne changerait en rien les conclusions du travail réalisé, le but étant de mettre en évidence l'importance du modèle dans la détermination d'une valeur optimale de température pour chaque data center actif.

8.1.2.2 Méthode de résolution approchée

Aussi, dans le cadre de cette thèse, plusieurs algorithmes de résolution approchée ont été développés. Toutefois, afin d'implémenter ces derniers, nous avons dû concevoir notre propre simulateur de l'environnement InterCloud, car il n'existe aucun outil permettant de reproduire fidèlement, en dehors des caractéristiques physiques des VMs et des besoins en trafic, le cadre utilisé dans notre modélisation, à savoir la puissance variable des ventilateurs, le comportement du système de refroidissement, de même que les contraintes de co-localisation. De plus, le problème traité, à notre humble avis, n'ayant jamais été modélisé auparavant, nous avons également dû générer les instances que nous avons utilisées pour les différentes simulations. Aussi, afin de mettre en évidence les performances des algorithmes développés, ces derniers ont été comparés aux résultats de la méthode exacte pour les instances de petite taille, cependant pour les problèmes plus difficiles, afin d'effectuer une comparaison équitable, nous avons eu à utiliser comme base d'évaluation des algorithmes connus que nous avions eu à adapter aux besoins du problème.

8.2 Analyse des résultats

Une rapide analyse de l'ensemble des résultats obtenus démontre que ces derniers sont plus que satisfaisants. L'une des premières contributions réside dans l'approche de modélisation utilisée. En effet, considérant que plusieurs acteurs jouent un rôle prépondérant dans la consommation énergétique et l'impact environnemental des data centers, nous avons proposé un modèle d'optimisation de l'empreinte carbone qui intègre l'ensemble de ces entités, tout en considérant les complexes interactions entre ces dernières. Le réalisme du modèle est aussi accentué par la présence de contraintes de QdS, lesquelles traduisent les exigences des utilisateurs en termes de performances et d'emplacement des applications à héberger. Cette démarche plutôt complexe, quoique réalisée dans un cadre de simulation très simple de placement statique des applications, jette les fondements pour une évaluation beaucoup plus précise de l'empreinte écologique des environnements Cloud, à savoir que le modèle développé peut être facilement adapté afin d'y intégrer des mécanismes beaucoup plus complexes tels la migration ou la variation des charges.

Quant aux résultats numériques, il n'est pas difficile d'en apprécier la qualité avec les chiffres qui l'illustrent parfaitement. En effet, comme présenté dans les quatre (4) chapitres précédents, le modèle développé permet des économies en Gaz à Effet de Serre (GES) pouvant aller de 30% jusqu'à 900%, comparées à différents modèles de base visant à atteindre un objectif similaire à celui défini dans cette thèse. Pour ce qui est des heuristiques développées, elles permettent de générer des solutions qui sont, en moyenne, de 0.2% à 3% de la solution optimale, pour un temps de calcul qui peut être jusqu'à 9 fois moins élevé que la durée d'exécution de l'algorithme de résolution exacte. Pour ce qui est des comparaisons avec d'autres méthodes approchées, les heuristiques proposées démontrent également d'excellentes performances, établissant ainsi un bon compromis entre la qualité de la solution et le temps d'exécution.

De manière générale, en raison des intéressantes possibilités que présente le Cloud Computing, il n'est pas surprenant qu'on assiste à une féroce compétition entre les différents fournisseurs désireux, à tout prix, de faire main basse sur une grande part du marché. Toutefois, à l'heure où la facture énergétique augmente considérablement en raison de la forte puissance de calcul installée dans les data centers et exploitée à outrance, les préoccupations environnementales dans le secteur de l'informatique se font de plus en plus pressantes, amenant ainsi à l'avènement de l'informatique éco-responsable. Ainsi, pour les fournisseurs d'infrastructure Cloud, l'enjeu n'est plus uniquement d'ordre économique, mais est également lié à l'image de leurs entreprises. Ainsi, les résultats obtenus sont très encourageants pour les fournisseurs de Cloud, car ils disposeront d'un outil leur permettant de bien planifier le processus de placement des applications, tout en réduisant leur impact environnemental.

En conclusion, dans le but d'atteindre les objectifs énoncés à la section 1.3, tout au long de la thèse, nous avons eu à effectuer des choix méthodologiques, certains motivés par diverses raisons, ou encore découlant de certains travaux antérieurs, dont les résultats ne figurent pas nécessairement dans cette thèse. Au regard de la qualité des résultats obtenus, il nous est possible d'admettre que les objectifs ont été atteints grâce aux trois grands travaux présentés dans ce document. Il en découle alors que la démarche méthodologique utilisée était suffisamment adaptée aux besoins du travail. Néanmoins, cette approche méthodologique n'étant pas totalement parfaite, il nous reviendra, au moment de conclure cette thèse, d'en présenter les limites et les potentielles avenues de recherche.

CHAPITRE 9 CONCLUSION ET RECOMMANDATIONS

Ce chapitre a pour but d'effectuer un récapitulatif des différents travaux réalisés dans le cadre de cette thèse. Dans un premier temps, il sera question de mettre en évidence les principales contributions de la thèse. Par la suite, les limitations de nos travaux seront exposées et finalement des recommandations pouvant faire l'objet de recherches futures seront proposées.

9.1 Sommaire des contributions de la thèse

Notre objectif principal consistait à concevoir un cadre de planification globale des machines virtuelles et du trafic associé dans la perspective d'une minimisation de l'empreinte carbone dans un environnement InterCloud. Cette planification globale vise à permettre à un fournisseur d'infrastructure physique de réaliser une optimisation conjointe, où les différents facteurs, influençant autant la consommation énergétique que l'empreinte carbone, seront simultanément considérés dans le processus de placement des applications, sous la contrainte des exigences des clients. Dans cette optique, cette thèse a su donner lieu à plusieurs contributions, aussi bien dans le monde de la planification des applications dans le Cloud, que dans le domaine des métaheuristiques :

1. Nous avons d'abord développé un modèle plus précis de consommation énergétique des applications simples à VM unique, qui améliore les modèles déjà existants en combinant la dissipation des châssis et du système de refroidissement. Nous y avons intégré le comportement dynamique des équipements par rapport à la température, la nature hétérogène des équipements, de même que leur profil énergétique. Ce modèle de consommation a, par la suite, été combiné au facteur d'émission des data centers afin de dégager un modèle d'empreinte carbone pour un ensemble de data centers géographiquement distribués et alimentés par des sources d'énergie différentes. De plus, une étude de la fonction objectif a été réalisée en regard de la température. Plus précisément, par le biais d'une analyse de la monotonie et de la convexité de la fonction objectif, nous avons pu mettre en évidence l'importance de déterminer la température optimale de chaque data center actif, soulignant ainsi la pertinence du modèle de consommation proposé.
2. Nous avons également proposé une nouvelle modélisation des exigences du client. En effet, contrairement aux travaux antérieurs qui associent les ententes de niveaux de services uniquement aux performances intrinsèques des applications hébergées, nous avons, dans notre travail, également considéré des requis visant à assurer la sécurité

et une certaine redondance des applications. Ainsi, notre travail fait état autant de la dégradation des performances des applications que des contraintes de co-localisation associées à ces dernières, telles qu'imposées par les clients.

3. Afin de réduire l'impact écologique d'un environnement InterCloud, nous avons, par le biais d'équations mathématiques, modélisé le problème de placement des applications à VM unique à l'échelle de plusieurs data centers, en combinant le modèle d'empreinte carbone proposé aux contraintes de co-hébergement et de performances des applications. Un modèle de programmation mathématique a été également proposé afin de résoudre, de manière exacte, le processus de placement et le comparer aux approches existantes. Plus spécifiquement, ce modèle permet à un fournisseur de déterminer l'emplacement idéal des VMs, de manière à réduire l'empreinte carbone de son infrastructure, tout en gardant comme point de mire la satisfaction des utilisateurs.
4. Nous avons développé une méthode de résolution approchée basée sur la Recherche Locale Itérée (ILS) afin de résoudre de plus grandes instances du problème, la méthode exacte étant limitée par la taille des jeux de données. La métaheuristique proposée tient compte autant de l'impact des nœuds de calcul que de celui du système de refroidissement de chaque data center, et implémente des mécanismes permettant de bien explorer l'espace des solutions. L'algorithme développé permet d'obtenir des résultats quasi optimaux et présente de bons résultats en termes de qualité de la solution et du temps d'exécution.
5. Un autre apport de la thèse consiste, par le biais d'équations mathématiques, à étendre le modèle présenté ci-haut, afin d'intégrer la prise en charge d'applications plus complexes dans le processus de placement. Dans la perspective d'une optimisation de l'empreinte carbone d'un environnement InterCloud, ce modèle nous permet de déterminer le serveur hôte pour chaque VM conjointement au routage efficace du trafic qui y est associé, sans oublier l'optimisation de l'efficacité énergétique. En ce sens, la consommation énergétique et l'impact environnemental des équipements réseau ont également été considérés.
6. Notre dernière réalisation, qui a été la conception d'un algorithme hybride, représente une contribution autant dans l'univers de la planification dans le Cloud que dans le domaine des métaheuristiques. En effet, dans la dernière phase de notre travail, nous avons proposé une méthode de résolution approximative basée sur l'utilisation combinée de deux métaheuristiques afin de résoudre le problème de planification globale des applications dans le Cloud. Plus spécifiquement, cette heuristique nous permet de considérer le problème de minimisation de l'empreinte carbone d'un InterCloud dans son intégralité, à savoir, simultanément définir l'emplacement des VMs, router le trafic

inter-VMs et maximiser l'efficacité du système de refroidissement dans chaque data center actif, toujours sous la contrainte des exigences des clients. De manière générale, cette méthode nous permet d'obtenir, en moyenne, des résultats quasi optimaux en un temps extrêmement réduit par rapport à la méthode exacte, pour les instances de petite taille. Pour les problèmes plus difficiles, une comparaison avec d'autres approches a pu démontrer que l'algorithme proposé procure un excellent compromis entre la qualité de la solution obtenue et la durée d'exécution. Dans le domaine des métaheuristiques, la contribution se rapporte au fait d'avoir adapté une approche peu utilisée, soit la combinaison de deux métaheuristiques, au problème de planification dans le Cloud. Cette hybridation consiste en une approche d'intégration où les traits caractéristiques des deux méthodes ont été largement exploités afin de faciliter une meilleure exploration de l'espace de recherche. Les résultats issus des comparaisons ont démontré que l'algorithme hybride proposé performe nettement mieux que chacune des métaheuristiques, prises une à une.

9.2 Limitations de la solution proposée

En dépit des nombreuses contributions énoncées ci-dessus, les travaux réalisés dans le cadre de cette thèse présentent certaines limitations.

— La principale limitation de cette thèse réside dans le fait que l'aspect dynamique associé à la planification dans le Cloud n'a pas été considéré. En effet, afin de mieux mettre en évidence les avantages du modèle d'optimisation de l'empreinte carbone proposé, nous avons dû considérer un cadre de planification exempt de toute dynamicité et de migration. Les approches actuellement utilisées ne considérant pas nécessairement tous les facteurs influençant la consommation énergétique d'un data center, le but visé par cette thèse était de démontrer que, même dans un scénario simple d'allocation statique des applications, une approche globale permettant de considérer simultanément ces différentes entités et les interactions complexes entre celles-ci, favorise le développement d'un modèle plus précis d'évaluation de l'empreinte carbone qui, lorsqu'utilisé dans le cadre d'une planification, permet d'obtenir une configuration de coût minimal. De ce choix découlent plusieurs autres tels la modélisation des charges à héberger, où les besoins en ressources physiques sont fixes et connus à l'avance, de même que le modèle de dégradation de performance utilisé. Toutefois, afin de mieux refléter la réalité dans le Cloud, il serait intéressant d'étudier comment cette dynamicité peut être intégrée au modèle actuel proposé.

— Au niveau de la modélisation des applications complexes, seul le cas des trafics inter-

VMs a été étudié. Un tel choix a été effectué dans le but d'intégrer le processus de routage du trafic et de l'évaluation de la consommation des ressources réseau, tout en maintenant un cadre simple de planification, et sans perte de généralité. Toutefois, il existe des applications qui font intervenir non seulement du trafic entre ses différentes VMs, mais qui nécessitent également des échanges de trafics entre des utilisateurs finaux et certaines de ses composantes. Il en découle que l'intégration de ce type d'interaction à la solution actuelle permettrait de mieux représenter les différents types d'applications susceptibles d'être hébergés dans le Cloud, élargissant ainsi l'éventail des utilisations possibles de l'approche proposée dans ce document.

— Une autre limitation associée au modèle développé est liée au fait d'avoir négligé tout phénomène de recirculation à l'intérieur d'un data center. Cette considération est, en général possible, lorsqu'en raison de la configuration interne du data center, le taux de recirculation de chaleur est quasi imperceptible, ou lorsque des séparateurs physiques sont installés de manière à isoler les couloirs d'air chaud, des rangées d'air froid. Toutefois, dans le cas où ces séparateurs physiques n'existeraient pas, il serait pertinent d'adapter le modèle de consommation énergétique de manière à prendre en considération ce phénomène de recirculation de chaleur, lequel influence grandement la température d'entrée des différents nœuds de calcul et l'efficacité du système de cooling.

— La discrétisation de la température aux valeurs entières a permis de linéariser le modèle mathématique afin de réduire la complexité du problème et d'accélérer le temps d'exécution de l'algorithme de résolution exacte. Toutefois, comme le niveau de granularité utilisée influence l'ensemble des valeurs de température à considérer dans un intervalle donné, nous estimons qu'une discrétisation plus fine des valeurs permettrait d'obtenir, sans doute, des valeurs plus précises d'empreinte carbone.

— Une autre limitation concerne la résolution exacte des modèles mathématiques. En effet, les modèles d'optimisation présentés aux chapitres 4 et 6 découlent de la programmation linéaire/non linéaire en nombres entiers et sont, par conséquent, très complexes à résoudre. En ce sens, très peu d'instances de petite taille, regroupant l'ensemble des contraintes du modèle, ont pu être résolues de manière exacte à l'aide du solveur CPLEX, comme l'illustrent les résultats de la section 7.1.2.

— Bien que le modèle d'empreinte carbone proposé considère toutes les contraintes liées autant aux équipements qu'aux exigences des utilisateurs, les méthodes de résolution approchées n'en intègrent qu'une partie, soit les contraintes de co-hébergement des VMs et de localisation des applications. Ce choix d'implémentation est intimement lié à la limitation énoncée précédemment. En effet, comme mentionné au chapitre 3,

les performances des métaheuristiques sont d'abord comparées aux résultats obtenus avec CPLEX, question de valider leur implémentation et le choix des paramètres de simulation. Toutefois, l'éventail des instances intégrant toutes les contraintes du modèle et pouvant être résolues de manière exacte étant très faible, il aurait été difficile d'apprécier la justesse de l'implémentation complète du modèle en considérant uniquement un nombre très restreint de jeux de données. De ce fait, un compromis s'imposait entre le nombre d'instances à tester et les contraintes à implémenter. Vu que l'une des contributions de la thèse consistait à intégrer la modélisation des exigences des clients en termes de sécurité et de redondance, nous avons délibérément choisi d'illustrer ce type de contraintes, au détriment des autres requis de performances des équipements et des applications, afin de considérer un plus large éventail d'instances à analyser.

9.3 Travaux futurs

Nous ne saurions conclure cette thèse sans faire état des pistes intéressantes pouvant faire l'objet de recherches futures. À ce stade, plusieurs voies de recherche, dont certaines découlent directement des limitations énoncées à la section précédente, s'offrent à nous.

— Une première piste à explorer serait de considérer la recirculation de la chaleur à l'intérieur d'un data center. Certains travaux se sont attaqués à l'étude de ce phénomène, dans le but de modéliser le processus de distribution de la température à l'intérieur d'un data center. Il serait alors intéressant d'étendre le modèle proposé afin d'intégrer et d'adapter les résultats issus de ces travaux antérieurs.

— Toujours dans le cadre de la modélisation, l'approche proposée considère que la température d'entrée des châssis est uniquement dépendante de l'air fourni par le système de refroidissement. Cependant, dans le contexte où un ou plusieurs data centers sont situés dans des régions où les ressources naturelles facilitent, à certaines périodes de l'année, le refroidissement gratuit, une avenue de recherche consisterait à évaluer les avantages à exploiter le *free cooling*, lorsque la température extérieure le permet, parallèlement à l'utilisation du CRAC.

— Du côté du processus de placement des applications, une voie importante à analyser est l'ajout du caractère dynamique au modèle. De manière générale, les besoins d'hébergement se manifestent à tout moment et les applications, même une fois hébergées, peuvent évoluer en termes de quantité de ressources physiques nécessaires. De ce fait, afin de mieux refléter la réalité des applications à héberger et du processus de placement de ces dernières, il serait intéressant de prendre en compte la notion d'incertitude associée aux charges à placer, d'adapter la modélisation des performances des applica-

tions en conséquence, et d'intégrer le concept de migration au modèle d'optimisation proposé.

— La prise en compte d'applications à multiples VMs, nous a portés à considérer une topologie en arbre comme schéma d'interconnexion entre les différents serveurs d'un data center, dans le but de réaliser le routage du trafic inter-VMs. Ainsi, il serait très attrayant d'implémenter le modèle proposé en considérant les autres topologies plus complexes. Plus spécifiquement, le but visé reviendrait à évaluer les performances du modèle en regard des différentes configurations utilisées.

— Une autre avenue de recherche très intéressante à explorer serait de proposer une approche de planification décentralisée, où une première phase de répartition des applications s'effectuerait à un plus haut niveau, et particulièrement du côté d'un contrôleur central, et où les décisions se prendraient en considérant la quantité de ressources agrégées de chaque data center. Une fois cette première phase complétée, les VMs et leurs trafics seraient répartis entre les data centers, et il reviendrait à des contrôleurs locaux, installés dans chaque data center, d'assigner les VMs aux bons serveurs et d'effectuer le routage du trafic associé. Le but ultime d'un tel travail de recherche consisterait à comparer les performances de l'approche distribuée à celles du modèle centralisé proposé dans cette thèse, en termes de qualité de la solution obtenue et du temps de convergence vers la solution optimale.

— Le développement de bornes inférieures serait également une piste intéressante à envisager. En effet, les performances des métaheuristiques développées n'ont pu être comparées à une borne inférieure, définie par les résultats obtenus avec la méthode exacte, que pour les instances de petite taille. Ceci est dû au fait qu'en raison de la complexité du modèle, les problèmes plus difficiles ne pouvaient être résolus de manière exacte. Ainsi, afin de mettre en évidence les performances des méthodes approchées pour les grandes instances, il nous a fallu concevoir des algorithmes qui, au terme des simulations, se sont plus révélés être des bornes supérieures, permettant de mettre en évidence les avantages du modèle proposé. Toutefois, ces résultats ne permettent pas nécessairement de conclure quant à la proximité de la solution obtenue avec la méthode proposée, par rapport à la valeur optimale inconnue. À cet effet, il serait plus qu'intéressant de déterminer un moyen de calculer des bornes inférieures de bonne qualité, particulièrement pour les instances de grande taille. Une idée serait de déterminer une équation mathématique pouvant être évaluée en un temps réduit. Grâce à l'évaluation d'une telle borne, il nous serait possible d'intégrer les contraintes de dégradation de performances des applications et de température de sortie des châssis dans nos métaheuristiques et d'en valider les résultats obtenus, ce qui n'a pas été

possible avec la méthode exacte.

— Il serait également captivant d'explorer une nouvelle forme d'hybridation, à savoir combiner une métaheuristique à une méthode exacte, dans le but de tirer profit des avantages des deux méthodes. Plus spécifiquement, une telle hybridation devrait permettre de réduire le temps de calcul exponentiel associé à la méthode exacte, tout en raffinant la solution sous-optimale retournée par la métaheuristique.

— Enfin, le point culminant de l'ensemble de ces travaux consisterait à développer un logiciel de planification des applications dans un environnement InterCloud. La mise en place de plusieurs interfaces usagers permettrait à tout fournisseur de spécifier les demandes en termes de VMs et trafics à héberger, de même que leurs caractéristiques, et compte tenu de la disponibilité au niveau de l'infrastructure physique, le programme déterminerait le schéma de configuration optimal permettant à ce fournisseur de minimiser l'empreinte carbone de son environnement Cloud.

RÉFÉRENCES

A VOUK, M. (2008). Cloud computing–issues, research and implementations. *CIT. Journal of Computing and Information Technology*, 16, 235–246.

ABDELSALAM, H. S., MALY, K., MUKKAMALA, R., ZUBAIR, M. et KAMINSKY, D. (2009). Analysis of energy efficiency in clouds. *Proceedings of the 2009 Computation World : Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns. COMPUTATIONWORLD'09*. IEEE Computer Society, 416–421.

AMAZON (2010). Amazon ec2. `https://aws.amazon.com/fr/ec2/`. Accédé le : 28/10/2015.

ANDERSON, E., HALL, J., HARTLINE, J., HOBBES, M., KARLIN, A., SAIA, J., SWAMINATHAN, R. et WILKES, J. (2010). Algorithms for data migration. *Algorithmica*, 57, 349–380.

ANDERSON, T., PETERSON, L., SHENKER, S. et TURNER, J. (2005). Overcoming the internet impasse through virtualization. *Computer*, 34–41.

ARMBRUST, M. *ET AL.* (2010). A view of cloud computing. *Communications of the ACM*, 53, 50–58.

ASHRAE, T. (2011). 9.9 (2011) thermal guidelines for data processing environments–expanded data center classes and usage guidance. *Whitepaper prepared by ASHRAE technical committee (TC)*, 9.

AVELAR, V., AZEVEDO, D., FRENCH, A. et POWER, E. N. (2012). Pue^{TM} : A comprehensive examination of the metric. *White paper*, 49.

BARROSO, L. A. et HÖLZLE, U. (2007). The case for energy-proportional computing. *IEEE computer*, 40, 33–37.

BELOGLAZOV, A., ABAWAJY, J. et BUYYA, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future generation computer systems*, 28, 755–768.

BELOGLAZOV, A. et BUYYA, R. (2010). Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers. *Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science*. ACM, Bangalore, India, 4.

BICHLER, M., SETZER, T. et SPEITKAMP, B. (2006). Capacity planning for virtualized servers. *Workshop on Information Technologies and Systems (WITS)*. Milwaukee, Wisconsin, USA.

BILAL, N., GALINIER, P. et GUIBAULT, F. (2014). An iterated-tabu-search heuristic for a variant of the partial set covering problem. *Journal of Heuristics*, 20, 143–164.

BIRAN, O., CORRADI, A., FANELLI, M., FOSCHINI, L., NUS, A., RAZ, D. et SILVERA, E. (2012). A stable network-aware vm placement for cloud systems. *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*. IEEE Computer Society, 498–506.

BOBROFF, N., KOCHUT, A. et BEATY, K. (2007). Dynamic placement of virtual machines for managing sla violations. *Proceedings of the 2007 10th IFIP/IEEE International Symposium on Integrated Network Management (IM'07)*. IEEE, 119–128.

BONDE, D. (2010). *Techniques for Virtual Machine Placement in Clouds*. Mémoire de maîtrise, Indian Institue of Technology.

BRUNETTE, G., MOGULL, R. *ET AL.* (2009). Security guidance for critical areas of focus in cloud computing v2. 1. *Cloud Security Alliance*, 1–76.

BURD, T. D., PERING, T. A., STRATAKOS, A. J. et BRODERSEN, R. W. (2000). A dynamic voltage scaled microprocessor system. *IEEE Journal of Solid-State Circuits*, 35, 1571–1580.

BUYYA, R., BELOGLAZOV, A. et ABAWAJY, J. (2010). Energy-efficient management of data center resources for cloud computing : a vision, architectural elements, and open challenges. *Proceedings of the 2010 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2010)*. CSREA Press, 6–17.

BUYYA, R., YEO, C. S., VENUGOPAL, S., BROBERG, J. et BRANDIC, I. (2009). Cloud computing and emerging it platforms : Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25, 599–616.

CARDOSA, M., KORUPOLU, M. R. et SINGH, A. (2009). Shares and utilities based power consolidation in virtualized server environments. *Proceedings of the 11th IFIP/IEEE international conference on Symposium on Integrated Network Management (IM'09)*. IEEE Press, Long Island, NY, 327–334.

CHAISIRI, S., LEE, B.-S. et NIYATO, D. (2009). Optimal virtual machine placement across multiple cloud providers. *2009 IEEE Asia-Pacific Services Computing Conference (APSCC)*. IEEE, 103–110.

CHEN, G., HE, W., LIU, J., NATH, S., RIGAS, L., XIAO, L. et ZHAO, F. (2008). Energy-aware server provisioning and load dispatching for connection-intensive internet services. *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX Association, vol. 8, 337–350.

CHOWDHURY, N., RAHMAN, M. et BOUTABA, R. (2009). Virtual network embedding with coordinated node and link mapping. *Proceedings of the 2009 IEEE International Conference on Computer Communications (INFOCOM 2009)*. IEEE, Rio De Janeiro, Brazil, 783–791.

CONGRAM, R. K., POTTS, C. N. et VAN DE VELDE, S. L. (2002). An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, 14, 52–67.

DHARWAR, D., BHAT, S. S., SRINIVASAN, V., SARMA, D. et BANERJEE, P. K. (2012). Approaches towards energy-efficiency in the cloud for emerging markets. *Proceedings of the 2012 IEEE International Conference on Cloud Computing in Emerging Markets, 2012 (CCEM 2012)*. IEEE, 1–6.

DONG, J., JIN, X., WANG, H., LI, Y., ZHANG, P. et CHENG, S. (2013). Energy-saving virtual machine placement in cloud data centers. *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid)*. IEEE, Delft, Netherlands, 618–624.

DOYLE, J., SHORTEN, R. et O'MAHONY, D. (2013). Stratus : Load balancing the cloud for carbon emissions control. *IEEE Transactions on Cloud Computing*, 1, 1–1.

DUPONT, C., GIULIANI, G., HERMENIER, F., SCHULZE, T. et SOMOV, A. (2012). An energy aware framework for virtual machine placement in cloud federated data centres. *Proceedings of the 3rd International Conference on Future Energy Systems : Where Energy, Computing and Communication Meet (e-Energy)*. IEEE, 1–10.

ESSEC (2009). Virtualisation, cloud computing et saas : Stratégies payantes en termes de roi ou miroirs aux alouettes ? `http://cr.g9plus.org/CR-241-2009-01-20.pdf`. Accédé le : 28/10/2015.

FAN, X., WEBER, W.-D. et BARROSO, L. A. (2007). Power provisioning for a warehouse-sized computer. *ACM SIGARCH Computer Architecture News*. ACM, vol. 35, 13–23.

FANG, S., KANAGAVELU, R., LEE, B.-S., FOH, C. H. et AUNG, K. M. M. (2013a). Power-efficient virtual machine placement and migration in data centers. *Proceedings of the 2013 IEEE International Conference on Green Computing and Communications (GreenCom'13) and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing (iThings/CPSCom'13)*. IEEE Computer Society, 1408–1413.

FANG, W., LIANG, X., LI, S., CHIARAVIGLIO, L. et XIONG, N. (2013b). Vmplanner : Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers. *Computer Networks*, 57, 179–196.

FELLER, E., RILLING, L. et MORIN, C. (2011). Energy-aware ant colony based workload placement in clouds. *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*. IEEE Computer Society, Washington, DC, USA, 26–33.

FOX, A., GRIFFITH, R., JOSEPH, A., KATZ, R., KONWINSKI, A., LEE, G., PATTERSON, D., RABKIN, A. et STOICA, I. (2009). Above the clouds : A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, 28, 13.

FUKUNAGA, A. S. (2009). Search spaces for min-perturbation repair. *Principles and Practice of Constraint Programming-CP 2009*, Springer. 383–390.

GAO, Y., GUAN, H., QI, Z., HOU, Y. et LIU, L. (2013). A multi-objective ant colony system algorithm for virtual machine placement in cloud computing. *Journal of Computer and System Sciences*, 79, 1230–1242.

GAREY, M. R. et JOHNSON, D. S. (1990). *Computers and Intractability ; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

GARG, S. K., YEO, C. S., ANANDASIVAM, A. et BUYYA, R. (2011a). Environment-conscious scheduling of hpc applications on distributed cloud-oriented data centers. *Journal of Parallel and Distributed Computing*, 71, 732–749.

GARG, S. K., YEO, C. S. et BUYYA, R. (2011b). Green cloud framework for improving carbon efficiency of clouds. *Proceedings of the 17th international conference on Parallel processing-Volume Part I (Euro-Par 2011)*. 491–502.

GOIRI, Í., HAQUE, M. E., LE, K., BEAUCHEA, R., NGUYEN, T. D., GUITART, J., TORRES, J. et BIANCHINI, R. (2015). Matching renewable energy supply and demand in green datacenters. *Ad Hoc Networks*, 25, 520–534.

GOZDECKI, J., JAJSZCZYK, A. et STANKIEWICZ, R. (2003). Quality of service terminology in ip networks. *IEEE Communications Magazine*, 41, 153–159.

GRANGE, P. (2010). Le livre blanc du cloud computing. *Paris : Syntec Informatique*.

GREENPEACE (2010). Make it green : Cloud computing and its contribution to climate change. Rapport technique, Greenpeace International.

HELLER, B., SEETHARAMAN, S., MAHADEVAN, P., YIAKOUMIS, Y., SHARMA, P., BANERJEE, S. et MCKEOWN, N. (2010). Elastictree : saving energy in data center networks. *Proceedings of the 7th USENIX conference on Networked systems design and implementation*. USENIX Association, 17–17.

HERMENIER, F., DEMASSEY, S. et LORCA, X. (2011). Bin repacking scheduling in virtualized datacenters. *Principles and Practice of Constraint Programming–CP 2011*, Springer. 27–41.

HERMENIER, F., LORCA, X., MENAUD, J.-M., MULLER, G. et LAWALL, J. (2009). Entropy : a consolidation manager for clusters. *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*. ACM, Washington, DC, USA, 41–50.

HICKS, P., WALNOCK, M. et OWENS, R. M. (1997). Analysis of power consumption in memory hierarchies. *Proceedings of the 1997 international symposium on Low power electronics and design*. ACM, 239–242.

HOELLER JR, A. S., WANNER, L. F. et FRÖHLICH, A. A. (2006). A hierarchical approach for power management on mobile embedded systems. *From Model-Driven Design to Resource Management for Distributed Embedded Systems*, Springer. 265–274.

HUA, S. et QU, G. (2003). Approaching the maximum energy saving on embedded systems with multiple voltages. *Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design*. IEEE Computer Society, 26.

HUA, S., QU, G. et BHATTACHARYYA, S. S. (2003). Energy reduction techniques for multimedia applications with tolerance to deadline misses. *Proceedings of the 40th annual Design Automation Conference*. ACM, 131–136.

HUANG, K., SANTINELLI, L., CHEN, J.-J., THIELE, L. et BUTTAZZO, G. C. (2010). Adaptive power management for real-time event streams. *Proceedings of the 2010 Asia and South Pacific Design Automation Conference*. IEEE Press, 7–12.

ISARD, M., BUDIU, M., YU, Y., BIRRELL, A. et FETTERLY, D. (2007). Dryad : distributed data-parallel programs from sequential building blocks. *ACM SIGOPS Operating Systems Review*. ACM, vol. 41, 59–72.

JIANG, J. W., LAN, T., HA, S., CHEN, M. et CHIANG, M. (2012). Joint vm placement and routing for data center traffic engineering. A. G. Greenberg et K. Sohraby, éditeurs, *Proceedings of the 2012 IEEE International Conference on Computer Communications (INFOCOM 2012)*. IEEE, 2876–2880.

JOHN, J. (2014). Green computing strategies for improving energy efficiency in it systems. *International Journal of Scientific Engineering and Technology*, 3, 715–717.

JUSTAFORT, V. D., BEAUBRUN, R. et PIERRE, S. (2014). A model for carbon footprint optimization in an intercloud environment. *Proceedings of the 2014 IEEE 3rd International Conference on Cloud Networking (CLOUDNET)*. IEEE, 426–431.

JUSTAFORT, V. D., BEAUBRUN, R. et PIERRE, S. (2015a). An iterated local search approach for carbon footprint optimization in an intercloud environment. *International Journal of Metaheuristics*. (forthcoming).

JUSTAFORT, V. D., BEAUBRUN, R. et PIERRE, S. (2015b). On the carbon footprint optimization in an intercloud environment. *IEEE Transactions on Cloud Computing*, PP, 1–1.

KAN, E. Y., CHAN, W. et TSE, T. (2010). Leveraging performance and power savings for embedded systems using multiple target deadlines. *Proceedings of the 2010 10th International Conference on Quality Software (QSIC)*. IEEE Computer Society, 473–480.

KANSAL, A., ZHAO, F., LIU, J., KOTHARI, N. et BHATTACHARYA, A. A. (2010). Virtual machine power metering and provisioning. *Proceedings of the 1st ACM symposium on Cloud computing*. ACM, Indianapolis, Indiana, USA, 39–50.

KHOSRAVI, A., GARG, S. K. et BUYYA, R. (2013). Energy and carbon-efficient placement of virtual machines in distributed cloud data centers. *Proceedings of the 19th international conference on Parallel Processing (Euro-Par 2013)*. 317–328.

KIM, S.-G., EOM, H. et YEOM, H. Y. (2013). Virtual machine consolidation based on interference modeling. *the journal of Supercomputing*, 66, 1489–1506.

KORD, N. et HAGHIGHI, H. (2013). An energy-efficient approach for virtual machine placement in cloud based data centers. *5th Conference on Information and Knowledge Technology, 2013 (IKT 2013)*. IEEE, 44–49.

KUSIC, D., KEPHART, J. O., HANSON, J. E., KANDASAMY, N. et JIANG, G. (2009). Power and performance management of virtualized computing environments via lookahead control. *Cluster computing*, 12, 1–15.

LAGHARI, K. U. R., CRESPI, N., MOLINA, B. et PALAU, C. E. (2011). Qoe aware service delivery in distributed environment. *Proceedings of the 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications (WAINA)*. IEEE Computer Society, 837–842.

LARUMBE, F. et SANSO, B. (2012). Cloptimus : A multi-objective cloud data center and software component location framework. *2012 IEEE 1st International Conference on Cloud Networking (CLOUDNET)*. IEEE, Paris, France, 23–28.

LARUMBE, F. et SANSO, B. (2013). A tabu search algorithm for the location of data centers and software components in green cloud computing networks. *IEEE Transactions on Cloud Computing*, 1, 22–35.

LEE, H. (2012). *An analysis of the impact of datacenter temperature on energy efficiency*. Thèse de doctorat, Massachusetts Institute of Technology.

LEE, Y. C. et ZOMAYA, A. Y. (2012). Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing*, 60, 268–280.

LEIVADEAS, A., PAPAGIANNI, C. et PAPAVASSILIOU, S. (2013). Efficient resource mapping framework over networked clouds via iterated local search-based request partitioning. *IEEE Transactions on Parallel and Distributed Systems*, 24, 1077–1086.

LENZEN, M. (2010). Current state of development of electricity-generating technologies : A literature review. *Energies*, 3, 462–591.

LI, B., LI, J., HUAI, J., WO, T., LI, Q. et ZHONG, L. (2009). Enacloud : An energy-saving application live placement approach for cloud computing environments. *2009 IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 17–24.

LI, D., CHOU, P. H. et BAGHERZADEH, N. (2002). Mode selection and mode-dependency modeling for power-aware embedded systems. *Proceedings of the 2002 Asia and South Pacific Design Automation Conference*. IEEE Computer Society, 697.

LI, K. (2015). Improving multicore server performance and reducing energy consumption by workload dependent dynamic power management. *IEEE Transactions on Cloud Computing*, PP, 1–1.

LIU, J., ZHAO, F., LIU, X. et HE, W. (2009). Challenges towards elastic power management in internet data centers. *Proceedings of the 2009 29th IEEE International Conference on Distributed Computing Systems Workshops*. IEEE Computer Society, 65–72.

LIU, Z., CHEN, Y., BASH, C., WIERMAN, A., GMACH, D., WANG, Z., MARWAH, M. et HYSER, C. (2012). Renewable and cooling aware workload management for sustainable data centers. *ACM SIGMETRICS Performance Evaluation Review*. ACM, vol. 40, 175–186.

LIU, Z., LIN, M., WIERMAN, A., LOW, S. et ANDREW, L. L. (2015). Greening geographical load balancing. *IEEE/ACM Transactions on Networking*, 23, 657–671.

LIU, Z., LIN, M., WIERMAN, A., LOW, S. H. et ANDREW, L. L. (2011). Geographical load balancing with renewables. *ACM SIGMETRICS Performance Evaluation Review*, 39, 62–66.

LORCH, J. R. et SMITH, A. J. (1998). Software strategies for portable computer energy management. *IEEE Personal Communications*, 5, 60–73.

MAHADEVAN, P., BANERJEE, S. et SHARMA, P. (2010). Energy proportionality of an enterprise network. *Proceedings of the first ACM SIGCOMM workshop on Green networking*. ACM, 53–60.

MAHADEVAN, P., SHARMA, P., BANERJEE, S. et RANGANATHAN, P. (2009). A power benchmarking framework for network devices. *Proceedings of the 8th International IFIP-TC 6 Networking Conference*. 795–808.

MANN, V., KUMAR, A., DUTTA, P. et KALYANARAMAN, S. (2011). Vmflow : leveraging vm mobility to reduce network power costs in data centers. *Proceedings of the 10th international IFIP TC 6 conference on Networking-Volume Part I (NETWORKING 2011)*. 198–211.

MAZZUCCO, M., DYACHUK, D. et DETERS, R. (2010). Maximizing cloud providers' revenues via energy aware allocation policies. *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD'10)*. IEEE Computer Society, Miami, Florida, USA, 131–138.

MCGEER, R., MAHADEVAN, P. et BANERJEE, S. (2010). On the complexity of power minimization schemes in data center networks. *2010 IEEE Global Telecommunications Conference (GLOBECOM 2010)*. IEEE, 1–5.

MENG, X., PAPPAS, V. et ZHANG, L. (2010). Improving the scalability of data center networks with traffic-aware virtual machine placement. *Proceedings of the 29th conference on Information communications (INFOCOM 2010)*. IEEE Press, 1–9.

MI, H., WANG, H., YIN, G., ZHOU, Y., SHI, D. et YUAN, L. (2010). Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers. *Proceedings of the 2010 IEEE International Conference on Services Computing (SCC 2010)*. IEEE Computer Society, 514–521.

MISHRA, M. et SAHOO, A. (2011). On theory of vm placement : Anomalies in existing methodologies and their mitigation using a novel vector based approach. *Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing (CLOUD)*. IEEE Computer Society, Washington, DC, USA, 275–282.

MITTAL, S. (2014). A survey of techniques for improving energy efficiency in embedded computing systems. *International Journal of Computer Aided Engineering and Technology*, 6, 440–459.

MIYOSHI, A., LEFURGY, C., VAN HENSBERGEN, E., RAJAMONY, R. et RAJKUMAR, R. (2002). Critical power slope : understanding the runtime effects of frequency scaling. *Proceedings of the 16th international conference on Supercomputing*. ACM, 35–44.

MOGHADDAM, F. F., CHERIET, M. et NGUYEN, K. K. (2011). Low carbon virtual private clouds. *Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing (CLOUD)*. IEEE Computer Society, Washington, DC, USA, 259–266.

MOGHADDAM, F. F., MOGHADDAM, R. F. et CHERIET, M. (2012). Carbon metering and effective tax cost modeling for virtual machines. *Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing (CLOUD'12)*. IEEE Computer Society, Honolulu, Hawaii, USA, 758–763.

MOORE, J. D., CHASE, J. S., RANGANATHAN, P. et SHARMA, R. K. (2005). Making scheduling" cool" : Temperature-aware workload placement in data centers. *USENIX annual technical conference, General Track*. Anaheim, CA, USA, 61–75.

MOSS, D. et BEAN, J. (2009). Energy impact of increased server inlet temperature. *APC White Paper*, 138.

MUSOLL, E. et CORTADELLA, J. (1995). Scheduling and resource binding for low power. *Proceedings of the 8th international symposium on System synthesis*. ACM, 104–109.

NAVET, N. et GAUJAL, B. (2006). Ordonnancement temps réel et minimisation de la consommation d'énergie. *Systèmes temps réel 2-Ordonnancement, réseaux et qualité de service*.

NEDEVSCHI, S., POPA, L., IANNACCONE, G., RATNASAMY, S. et WETHERALL, D. (2008). Reducing network energy consumption via sleeping and rate-adaptation. *NSDI*. vol. 8, 323–336.

PAKBAZNIA, E. et PEDRAM, M. (2009). Minimizing data center cooling and server power costs. *Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design*. ACM, San Fancisco, CA, USA, 145–150.

PARAIN, F., BANÂTRE, M., CABILLIC, G., HIGUERA, T., ISSARNY, V. et LESOT, J.-P. (2000). Techniques de réduction de la consommation dans les systèmes embarqués temps-réel.

PAROLINI, L., SINOPOLI, B. et KROGH, B. H. (2008). Reducing data center energy consumption via coordinated cooling and load management. *Proceedings of the 2008 conference on Power aware computing and systems, HotPower*. vol. 8, 14–14.

PATEL, C. D., BASH, C. E., SHARMA, R., BEITELMAL, M. et FRIEDRICH, R. (2003). Smart cooling of data centers. *ASME 2003 International Electronic Packaging Technical Conference and Exhibition*. American Society of Mechanical Engineers, Maui, Hawaii, USA, 129–137.

PELLEY, S., MEISNER, D., WENISCH, T. F. et VANGILDER, J. W. (2009). Understanding and abstracting total data center power. *Workshop on Energy-Efficient Design*.

PENG, J., ZHANG, X., LEI, Z., ZHANG, B., ZHANG, W. et LI, Q. (2009). Comparison of several cloud computing platforms. *Proceedings of the 2009 Second International Symposium on Information Science and Engineering (ISISE)*. IEEE Computer Society, 23–27.

PILLEMENT, S., DAVID, R. et SENTIEYS, O. (2003). Architectures reconfigurables : opportunités pour la faible consommation. *papier invité aux journées Faible Tension Faible Consommation (FTFC)*.

POLVERINI, M., CIANFRANI, A., REN, S. et VASILAKOS, A. V. (2014). Thermal-aware scheduling of batch jobs in geographically distributed data centers. *IEEE Transactions on Cloud Computing*, 2, 71–84.

POSLADEK, G. (2008). *An investigation into using free cooling and community heating to reduce data centre energy consumption*. Mémoire de maîtrise, Strathclyde University.

QUAN, D. M., BASMADJIAN, R., DE MEER, H., LENT, R., MAHMOODI, T., SANNELLI, D., MEZZA, F., TELESCA, L. et DUPONT, C. (2012). Energy efficient resource allocation strategy for cloud data centres. *Computer and Information Sciences II*, Springer. 133–141.

QUAN, G. et HU, X. (2001). Energy efficient fixed-priority scheduling for real-time systems on variable voltage processors. *Proceedings of the 38th annual Design Automation Conference*. IEEE, 828–833.

RASMUSSEN, N. (2007). Calculating total cooling requirements for data centers. *White Paper*, 25, 1–8.

ROUSSEAU, L.-M. et PESANT, G. (2005). Programmation par contraintes. *Gestion de production et ressources humaines : méthodes de planification dans les systèmes productifs*, 223.

SENTIEYS, O. (1997). Réduction de consommation d'énergie en électronique embarquée. *Journée scientifique électronique embarquée du*, 24.

SHANG, Y., LI, D. et XU, M. (2010). Energy-aware routing in data center network. *Proceedings of the first ACM SIGCOMM workshop on Green networking*. ACM, 1–8.

SHARIFI, M., SALIMI, H. et NAJAFZADEH, M. (2012). Power-efficient distributed scheduling of virtual machines using workload-aware consolidation techniques. *The Journal of Supercomputing*, 61, 46–66.

SHARMA, R. K., BASH, C. E., PATEL, C. D., FRIEDRICH, R. J. et CHASE, J. S. (2005). Balance of power : Dynamic thermal management for internet data centers. *IEEE Internet Computing*, 9, 42–49.

SHUJA, J., BILAL, K., MADANI, S. A. et KHAN, S. U. (2014). Data center energy efficient resource scheduling. *Cluster Computing*, 17, 1265–1277.

SPEITKAMP, B. et BICHLER, M. (2010). A mathematical programming approach for server consolidation problems in virtualized data centers. *IEEE Transactions on Services Computing*, 3, 266–278.

SRIKANTAIAH, S., KANSAL, A. et ZHAO, F. (2008). Energy aware consolidation for cloud computing. *Proceedings of the 2008 conference on Power aware computing and systems*. San Diego, California, vol. 10.

ST-HILAIRE, M., CHAMBERLAND, S. et PIERRE, S. (2006). Uplink umts network design—an integrated approach. *Computer Networks*, 50, 2747–2761.

SU, C.-L. et DESPAIN, A. M. (1995). Cache designs for energy efficiency. *Proceedings of the 28th Hawaii International Conference on System Sciences*. IEEE Computer Society, vol. 1, 306–315.

TANG, M. et PAN, S. (2014). A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers. *Neural Processing Letters*, 41, 211–221.

TANG, Q., MUKHERJEE, T., GUPTA, S. K. et CAYTON, P. (2006). Sensor-based fast thermal evaluation model for energy efficient high-performance datacenters. *2006 Fourth International Conference on Intelligent Sensing and Information Processing (ICISIP 2006)*. IEEE, Bangalore, India, 203–208.

VAN, H. N., TRAN, F. D. et MENAUD, J.-M. (2010). Performance and power management for cloud infrastructures. *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE Computer Society, 329–336.

VAN HEDDEGHEM, W., VEREECKEN, W., COLLE, D., PICKAVET, M. et DEMEESTER, P. (2012). Distributed computing for carbon footprint reduction by exploiting low-footprint energy availability. *Future Generation Computer Systems*, 28, 405–414.

VERMA, A., AHUJA, P. et NEOGI, A. (2008a). pmapper : power and migration cost aware application placement in virtualized systems. *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware (Middleware 2008)*. 243–264.

VERMA, A., AHUJA, P. et NEOGI, A. (2008b). Power-aware dynamic placement of hpc applications. *Proceedings of the 22nd annual international conference on Supercomputing*. ACM, 175–184.

VON LASZEWSKI, G., WANG, L., YOUNGE, A. J. et HE, X. (2009). Power-aware scheduling of virtual machines in dvfs-enabled clusters. *Proceedings of the 2009 IEEE International Conference on Cluster Computing and Workshops, 2009 (CLUSTER'09)*. IEEE, 1–10.

VU, H. T. et HWANG, S. (2014). A traffic and power-aware algorithm for virtual machine placement in cloud data center. *International Journal of Grid & Distributed Computing*, 7, 350–355.

WADHWA, B. et VERMA, A. (2014). Energy and carbon efficient vm placement and migration technique for green cloud datacenters. *2014 Seventh International Conference on Contemporary Computing (IC3)*. IEEE, 189–193.

WILLIAMS, H. P. (2013). *Model building in mathematical programming*. John Wiley & Sons, Chichester, New York, Brisbane.

WOOD, T., SHENOY, P., VENKATARAMANI, A. et YOUSIF, M. (2009). Sandpiper : Black-box and gray-box resource management for virtual machines. *Computer Networks*, 53, 2923–2938.

WU, G., TANG, M., TIAN, Y.-C. et LI, W. (2012). Energy-efficient virtual machine placement in data centers by genetic algorithm. *Proceedings of the 19th international conference on Neural Information Processing-Volume Part III*. Springer-Verlag, 315–323.

WU, Y. (2013). *Energy efficient virtual machine placement in data centers*. Mémoire de maîtrise, Queensland University of Technology.

XIE, R., JIA, X., YANG, K. et ZHANG, B. (2013). Energy saving virtual machine allocation in cloud computing. *Proceedings of the 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops (ICDCSW)*. IEEE Computer Society, 132–137.

XU, J. et FORTES, J. A. (2010). Multi-objective virtual machine placement in virtualized data center environments. *Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications (GreenCom'10) & Int'l Conference on Cyber, Physical and Social Computing (CPSCom'10)*. IEEE Computer Society, 179–188.

XU, L., ZENG, Z. et YE, X. (2012). Multi-objective optimization based virtual resource allocation strategy for cloud computing. *Proceedings of the 2012 IEEE/ACIS 11th International Conference on Computer and Information Science*. IEEE Computer Society, 56–61.

ZHAI, B., DRESLINSKI, R. G., BLAAUW, D., MUDGE, T. et SYLVESTER, D. (2007). Energy efficient near-threshold chip multi-processing. *Proceedings of the 2007 international symposium on Low power electronics and design*. ACM, 32–37.

ZHENG, K., WANG, X., LI, L. et WANG, X. (2014). Joint power optimization of data center network and servers with correlation analysis. *Proceedings of the 2014 IEEE International Conference on Computer Communications (INFOCOM 2014)*. IEEE, 2598–2606.

ZHU, Y. et AMMAR, M. (2006). Algorithms for assigning substrate network resources to virtual network components. *Proceedings of the 2006 IEEE 25th International Conference on Computer Communications (INFOCOM 2006)*. IEEE, 1–12.